FLÁVIO SZENBERG

ACOMPANHAMENTO DE CENAS COM CALIBRAÇÃO AUTOMÁTICA DE CÂMERAS

TESE DE DOUTORADO

DEPARTAMENTO DE INFORMÁTICA

Rio de Janeiro, 19 de dezembro de 2001

FLÁVIO SZENBERG

ACOMPANHAMENTO DE CENAS COM CALIBRAÇÃO AUTOMÁTICA DE CÂMERAS

TESE DE DOUTORADO

DEPARTAMENTO DE INFORMÁTICA

PUC-Rio

Rio de Janeiro, 19 de dezembro de 2001

FLÁVIO SZENBERG

ACOMPANHAMENTO DE CENAS COM CALIBRAÇÃO AUTOMÁTICA DE CÂMERAS

Tese apresentada ao Departamento de Informática da PUC-Rio como parte dos requisitos para a obtenção do título de Doutor em Ciências em Informática.

Orientador: Marcelo Gattass

Co-orientador: Paulo Cezar Pinto Carvalho

Departamento de Informática

Pontifícia Universidade Católica do Rio de Janeiro

Rio de Janeiro, 19 de dezembro de 2001

À Minha Família

Agradecimentos

À minha esposa Solange Stern Szenberg pela compreensão, força, apoio, ajuda e incentivo durante todo o tempo e pela falta de tempo para passear.

À minha filha Rachel Elisa Szenberg pelas diversas vezes em que não foi possível brincar com ela, tendo que dedicar-me ao trabalho, e por ceder, de livre e espontânea vontade minha, sua imagem para os testes realizados nesta tese.

Ao meu pai Jak Szenberg, minha mãe Cheiva Szenberg, minha irmã Ana Sarah, meus cunhados Simão e Ricardo, meus sogros Samuel e Aída e meus sobrinhos Rosa Malca e Samuel Moshé, pela força e apoio que me deram para realizar esta tese.

Aos orientadores, Prof. Paulo Cezar Pinto Carvalho e Prof. Marcelo Gattass, pela ajuda, incentivo, orientação e as valiosas contribuições ao longo da construção desta tese e também pela confiança depositada em meu trabalho.

Aos amigos e companheiros do TeCGraf/PUC-Rio, Visgraf/IMPA e do Departamento de Informática da PUC-Rio que contribuíram direta ou indiretamente para a realização deste trabalho.

Ao CNPq e PROTEM/CC-GEOTEC pelo apoio financeiro.

Resumo

É cada vez mais comum, na transmissão de eventos esportivos pelas emissoras de televisão, a inserção, em tempo real, de elementos sintéticos na imagem, como anúncios, marcações no campo, etc. Geralmente, essa inserção é feita através do emprego de câmeras especiais, previamente calibradas e dotadas de dispositivos que registram seu movimento e a mudança de seus parâmetros. De posse destas informações, é simples inserir novos elementos na cena com a projeção apropriada.

Nesta tese, é apresentado um algoritmo para recuperar, em tempo real e sem utilizar qualquer informação adicional, a posição e os parâmetros da câmera em uma seqüência de imagens contendo a visualização de modelos conhecidos. Para tal, é explorada a existência, nessas imagens, de segmentos de retas que compõem a visualização do modelo cujas posições são conhecidas no mundo tridimensional. Quando se trata de uma partida de futebol, por exemplo, o modelo em questão é composto pelo conjunto das linhas do campo, segundo as regras que definem sua geometria e dimensões.

Inicialmente, são desenvolvidos métodos para a extração de segmentos de retas longos da primeira imagem. Em seguida é localizada uma imagem do modelo no conjunto desses segmentos com base em uma árvore de interpretação. De posse desse reconhecimento, é feito um reajuste nos segmentos que compõem a visualização do modelo, sendo obtidos pontos de interesse que são repassados a um procedimento capaz de encontrar a câmera responsável pela visualização do modelo. Para a segunda imagem da seqüência em diante, apenas uma parte do algoritmo é utilizada, levando em consideração a coerência entre quadros, a fim de aumentar o desempenho e tornar possível o processamento em tempo real.

Entre diversas aplicações que podem ser empregadas para comprovar o desempenho e a validade do algoritmo proposto, está uma que captura imagens através de uma câmera para demonstrar o funcionamento do algoritmo *on line*. A utilização de captura de imagens permite testar o algoritmo em inúmeros casos, incluindo modelos e ambientes diferentes.

Abstract

In the television casting of sports events, it has become very common to insert synthetic elements to the images in real time, such as adds, marks on the field, etc. Usually, this insertion is made using special cameras, previously calibrated and provided with features that record their movements and parameter changes. With such information, inserting new objects to the scene with the adequate projection is a simple task.

In the present work, we will introduce an algorithm to retrieve, in real time and using no additional information, the position and parameters of the camera in a sequence of images containing the visualization of previously-known models. For such, the method explores the existence in these images of straight-line segments that compose the visualization of the model whose positions are known in the three-dimensional world. In the case of a soccer match, for example, the respective model is composed by the set of field lines determined by the rules that define their geometry and dimensions.

Firstly, methods are developed to extract long straight-line segments from the first image. Then an image of the model is located in the set formed by such segments based on an interpretation tree. With such information, the segments that compose the visualization of the model are readjusted, resulting in the obtainment of interest points which are then passed to a proceeding able to locate the camera responsible for the model's visualization. For the second image on, only a part of the algorithm is used, taking into account the coherence between the frames, with the purpose of improving performance to allow real-time processing.

Among several applications that can be employed to evaluate the performance and quality of the proposed method, there is one that captures images with a camera to show the on-line functioning of the algorithm. By using image capture, we can test the algorithm in a great variety of instances, including different models and environments.

Sumário

Índice de Figuras

Índice de Tabelas

Lista de Notações e Símbolos

S.....: seqüência de imagens;

I_t .....: imagem no tempo t do vídeo;

fps.....: "*frames*" por segundo ou quadros por segundo;

(u, v).....: coordenada de um ponto da imagem;

(u_0, v_0) .....: coordenada do centro da imagem;

$(\widetilde{u}, \widetilde{v})$.....: coordenada de um ponto da imagem deslocado de $-(u_0, v_0)$;

$(\overline{u}, \overline{v})$.....: coordenada do centróide de uma célula;

(x, y, z).....: coordenada de um ponto da cena no sistema do mundo;

p_i.....: i-ésimo ponto da imagem;

P_i .....: i-ésimo ponto do mundo da cena;

(P_i, p_i) .....: i-ésimo par de pontos para a calibração de câmera ou estimação de uma homografia;

H.....: transformação projetiva planar ou homografia;

∇.....: operador matemático laplaciano;

LoG .....: filtro laplaciano da gaussiana;

θ.....: inclinação de uma reta, correspondente à equação (3.1);

ρ.....: distância de uma reta à origem, correspondente à equação (3.1);

λ_1, λ_2 .....: autovalores da matriz de covariância;

ν_{λ_1} , ν_{λ_2} .....: autovetores da matriz de covariância relativos aos autovalores λ_1 e λ_2, respectivamente;

κ_i.....: peso ou valor do ponto (u_i, v_i);

F_i.....: i-ésimo segmento de reta do modelo real para reconhecimento;

f_i .....: i-ésimo segmento de reta da visualização para reconhecimento;

$f_x : F_y$ .....: linha f_x da visualização representa a linha F_y do modelo;

$\{F_{i1}, F_{i2}, \ldots, F_{in}\}$.: solução do processo de reconhecimento da visualização – significa $f_1 : F_{i1}$, $f_2 : F_{i2}$, …, $f_n : F_{in}$;

$\varnothing$ .....: não representatividade no modelo;

η .....: fator de distorção perspectiva dada na direção $\overrightarrow{ov}$;

ρ .....: tolerância de paralelismo;

Capítulo 1

1 Introdução

Recentemente, diversas emissoras de televisão vêm utilizando recursos de computação gráfica para que os telespectadores tenham mais interesse em assistir eventos esportivos, obtendo e inserindo informações à cena através da combinação de imagens virtuais com imagens reais. Em alguns eventos esportivos, tais como jogos de futebol, muitas vezes os comentaristas utilizam esses recursos para fornecer informações mais precisas sobre o jogo, como, por exemplo, para verificar o impedimento de jogadores ou se a barreira está na distância correta, como ilustrado na Figura 1.1.

Figura 1.1 – Verificação da distância da barreira – extraída de [ORAD].

Outro exemplo de recurso utilizado pelas emissoras é a inserção de propagandas comerciais ou outros tipos de informações, como mostrado na Figura 1.2. Neste exemplo, foram colocadas algumas bandeiras nacionais (virtuais) no gramado, sobre um desfile, informando qual país está desfilando no momento.

Figura 1.2 – Bandeiras virtuais inseridas em uma cena – extraída de [ORAD].

Para que a inserção de objetos virtuais em imagens reais possa ser realizada, é necessário conhecer a câmera, obtendo informações como sua posição, orientação e inclinação (parâmetros extrínsecos), além de características de seu sistema óptico (parâmetros intrínsecos). A partir destas informações e das posições projetadas dos objetos na cena de pontos da imagem real, é possível fazer medições, como ilustra a Figura 1.1. Muitas vezes, quando todos os elementos de interesse estão em um único plano, basta encontrar uma transformação projetiva planar para atender a esses objetivos. Então, o problema de inserir objetos virtuais em imagens reais pode ser colocado como um problema de determinação dos parâmetros da câmera e sobreposição ou combinação de imagens – imagens reais mais imagens virtuais.

Uma vez conhecida a câmera, podemos modelar a cena 3D a partir da imagem, como vemos na Figura 1.3, onde a câmera é representada por um círculo azul. A sombra no gramado do modelo virtual representa a região do gramado que é visualizada na imagem real, ilustrada no retângulo com bordas pretas na forma de um telão.

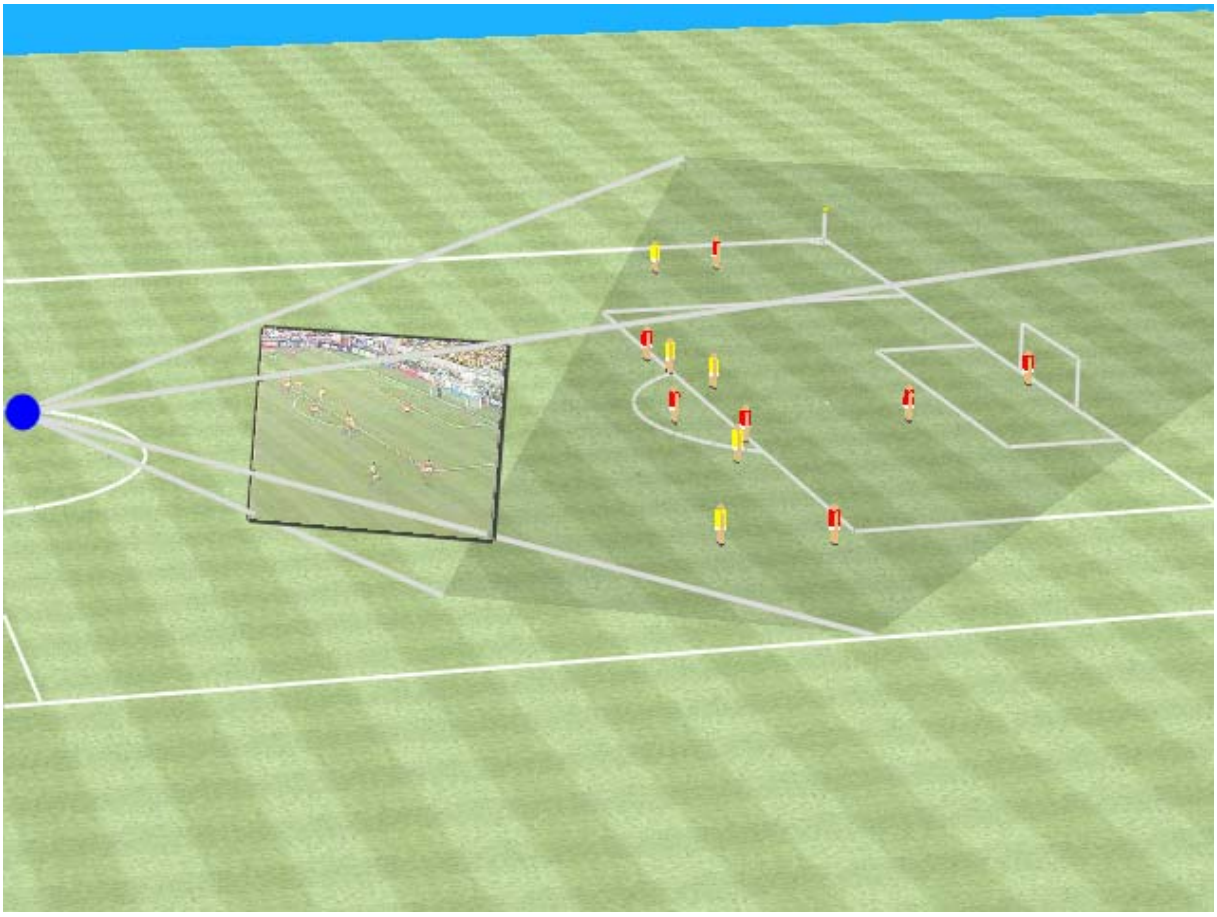


Figura 1.3 – Uma vez conhecida a câmera, a cena 3D pode ser modelada.

1.1 Sistemas Existentes

Existe um grande número de sistemas comerciais disponíveis no mercado com a finalidade de extrair informações de imagens e inserir objetos virtuais. Na maioria desses sistemas, o cenário em questão diz respeito a esportes, como, por exemplo, uma partida de futebol. Alguns exemplos desses sistemas comerciais, utilizados por diversas emissoras de televisão, são:

- *VirtuaLive* [VirtuaLive]: permite criar animações em formato VRML a partir de uma seqüência de vídeo curta de uma partida de futebol. Os jogadores são representados por polígonos.
- *1st&Ten* [Firstandten]: projetado para inserir uma linha virtual no gramado, em transmissões ao vivo de partidas de futebol americano. Em [Firstandten] é detalhado seu mecanismo de funcionamento com relação tanto ao equipamento,

que utiliza sensores nas câmeras, quanto ao algoritmo. O seu esquema está ilustrado na Figura 1.4.

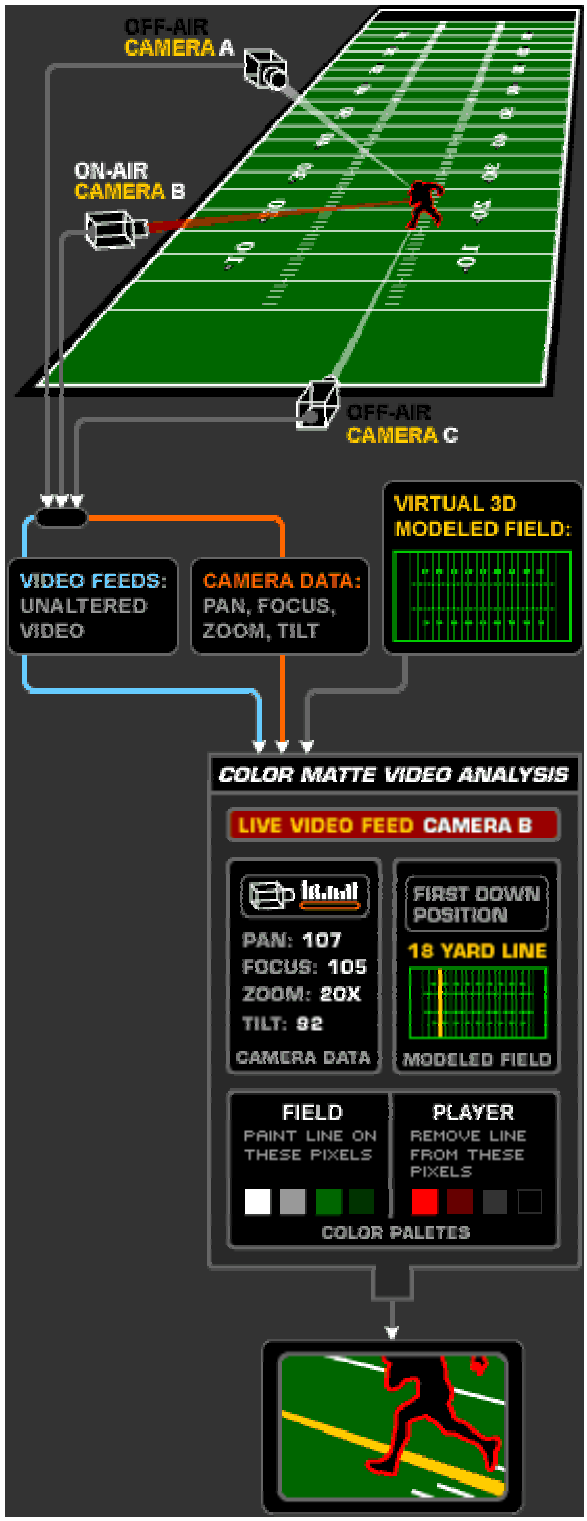


Figura 1.4 – Esquema de funcionamento do sistema 1st&Ten – extraída de [Firstandten].

- *Digital Replay* [DigitalReplay]: realiza o acompanhamento de atletas em uma seqüência de imagens, podendo, entre outras coisas, calcular medidas em um campo de futebol, determinar a velocidade dos jogadores e da bola e mostrar a linha de impedimento em transmissões de partidas de futebol.
- *VirtualReplay* [VirtualReplay]: produz uma reconstrução 3D de uma imagem de um jogo de futebol. Uma interação manual é necessária para criar os jogadores, que são representados por polígonos, utilizando tecnologia de *bill board*. Como o sistema opera com uma imagem tomada de uma única câmera, não é possível calcular a posição 3D da bola e dos jogadores quando não estão na altura do gramado.
- Outros sistemas usados comercialmente são *PicthTrax* [PicthTrax], *TennisProView* [TennisProView] e *GolfProView* [GolfProView], que calculam a trajetória, em coordenadas do mundo da cena, das bolas de beisebol, tênis e golfe, respectivamente.

Todos estes recursos são realizados e aplicados utilizando-se a câmera responsável pela visualização da cena como sendo a mesma para desenhar os objetos virtuais através de técnicas de computação gráfica. Para isto, conforme pode ser visto na Figura 1.5, existem dois meios de se realizar a combinação de imagens: através de acompanhamento de câmera ou através de calibração de câmera.

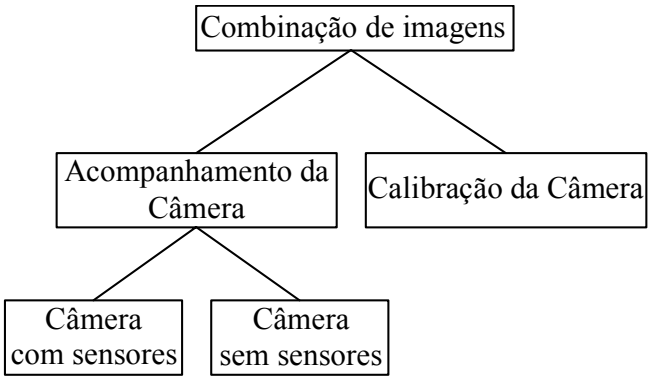


Figura 1.5 – Esquema para combinação de imagens.

No método de acompanhamento de câmera, a câmera é calibrada apenas na primeira imagem da seqüência de vídeo, a partir da localização do padrão conhecido contido nela. Para as demais imagens da seqüência, os parâmetros da câmera são reajustados diretamente a

partir do deslocamento do padrão na imagem. Por outro lado, quando a combinação de imagens é feita através da calibração de câmera, o padrão é localizado sempre em cada imagem e, a partir daí, são encontrados os parâmetros da nova câmera. Isto é, no acompanhamento de câmera, é encontrada somente uma câmera para todas as imagens, apenas atualizando seus parâmetros, enquanto na calibração de câmera uma nova câmera é encontrada a cada imagem.

Quando a combinação de imagens é feita por meio de acompanhamento de câmera, ele pode ser realizado através de duas técnicas:

- via utilização de sensores, incorporando equipamentos às câmeras capazes de notificar ao sistema as alterações ocorridas em seus parâmetros intrínsecos e extrínsecos;
- sem a utilização de sensores, calculando apenas com base na imagem corrente e na imagem anterior os novos parâmetros da câmera.

Para fazer a combinação de imagens usando calibração de câmera, pressupõe-se que é feita alguma detecção de um determinado padrão conhecido nas imagens reais. A detecção pode ser realizada através da intervenção de uma pessoa, informando pontos que caracterizam localização do padrão na imagem e seus correspondentes no padrão real, ou sem esta intervenção, com o sistema localizando e reconhecendo automaticamente o padrão desejado na imagem.

Todos os sistemas comerciais listados acima utilizam métodos de acompanhamento de câmera, como pode ser observado em suas referências e na Figura 1.4, que ilustra o esquema de funcionamento do sistema 1st&Ten.

No entanto, no caso do emprego desses sistemas pelas emissoras de televisão, o processo de acompanhamento de câmera é facilitado pois elas possuem as câmeras, podendo extrair suas propriedades e características diretamente por intermédio de equipamentos acoplados à câmera, como sensores de movimento. Porém, quando deseja-se inserir objetos virtuais em uma seqüência de imagens, ou vídeo, já produzida ou extraída por meios de captura, que é o objetivo desta tese, não existe essa facilidade. Outro problema é que esses sensores são muitas vezes difíceis de serem instalados em uma câmera comum e seu custo é geralmente elevado.

Por isso, esta tese se propõe a resolver o problema da combinação de imagens sem a utilização de sensores. Para tanto, faz-se necessário buscar meios de encontrar os parâmetros da câmera para todas as imagens do vídeo para poder então inserir os objetos virtuais desejados em posições corretas em cada imagem, seja por acompanhamento ou por calibração de câmera. Além disso, mesmo contendo sensores nas câmeras, todos os sistemas utilizados pelas emissoras de televisão exigem uma calibração inicial da câmera – os sensores notificam apenas alterações sofridas nos parâmetros.

1.2 Descrição do Problema

Conforme mencionado na seção anterior, o problema a ser resolvido nesta tese pode ser colocado da seguinte forma:

> *Dada uma seqüência de imagens que apresentam a visualização, total ou parcial, de um determinado modelo, acompanhar este modelo, calibrando as câmeras para cada imagem de forma automática, a fim de sobrepor objetos virtuais*.

O algoritmo proposto nesta tese procura resolver este problema visando obter a menor intervenção possível do usuário no que diz respeito a fornecer algum tipo de dado ou parâmetros externos. É também nosso objetivo buscar algoritmos que quando implementados sejam capazes de processar cerca de 30 quadros de uma seqüência por segundo para permitir sua utilização em tempo real.

1.3 Algoritmo Proposto

Um meio de encontrar os parâmetros da câmera é tentar acompanhar um determinado padrão conhecido na cena da visualização. Com as coordenadas do padrão na cena, é possível determinar os parâmetros da câmera, como é visto no algoritmo proposto nesta tese. O padrão, neste caso, é dado por um modelo descrito por um conjunto de segmentos de retas paralelos ou ortogonais entre si. Quando se trata de uma partida de futebol, o padrão é o conjunto das linhas retas do campo de futebol.

O algoritmo proposto realiza um acompanhamento de um modelo seguido de um método de calibração de câmera ([Szenberg+2001]). Este algoritmo enquadra-se na categoria

de combinação de imagens através de calibração de câmera, segundo o esquema da Figura 1.5.

O algoritmo proposto nesta tese é um método novo de acompanhamento de cena que realiza uma calibração de câmera para cada imagem de uma seqüência dada em tempo real. Nesta tese são descritas e utilizadas, para o método proposto, diversas técnicas de processamento de imagens, reconhecimento de padrões, computação gráfica e visão computacional, que podem ser listadas da seguinte forma:

- Filtragem e detecção de linhas: são discutidos alguns métodos descritos na literatura para detectar pontos passíveis de estarem sobre linhas presentes em imagens e filtros para suavizar problemas presentes na imagem, como, por exemplo, ruídos;
- Extração de segmentos de reta longos: é apresentado um método novo para a extração de segmentos longos e finos e comparado com o método tradicional baseado na transformada de Hough;
- Reconhecimento da visualização: um método de reconhecimento baseado no modelo descrito em [Grimson90] é ligeiramente modificado e aprimorado para o modelo que desejamos acompanhar;
- Mapeamento projetivo planar: são descritas duas técnicas para encontrar uma transformação projetiva planar para reconstruir o modelo, que são analisadas individualmente e comparadas entre si;
- Reconstrução da visualização do modelo: usando transformação projetiva, a visualização do modelo pode ser feita sobrepondo à existente na imagem. Uma técnica simples para a substituição dessa reconstrução é discutida, substituindo o modelo extraído da imagem.
- Acompanhamento em uma seqüência de imagens: é apresentada uma nova técnica simples de acompanhamento de padrões em uma seqüência sem cortes secos fazendo reajustes de posições, isto é, onde existem coerências entre imagens consecutivas da seqüência, e são enumeradas técnicas de estimação em quadros futuros;

- Calibração de câmeras: são apresentados dois algoritmos de calibração de câmeras, sendo um clássico, chamado Método de Tsai [Tsai86], e o outro implementado no sistema Juiz Virtual [JuizVirtual].

Em muitas aplicações na área de análise de imagens, desejamos interpretar uma visualização a partir da extração de objetos conhecidos, conforme o esquema ilustrado na Figura 1.6. O algoritmo proposto nesta tese é derivado de um processo de análise de imagens desta categoria.

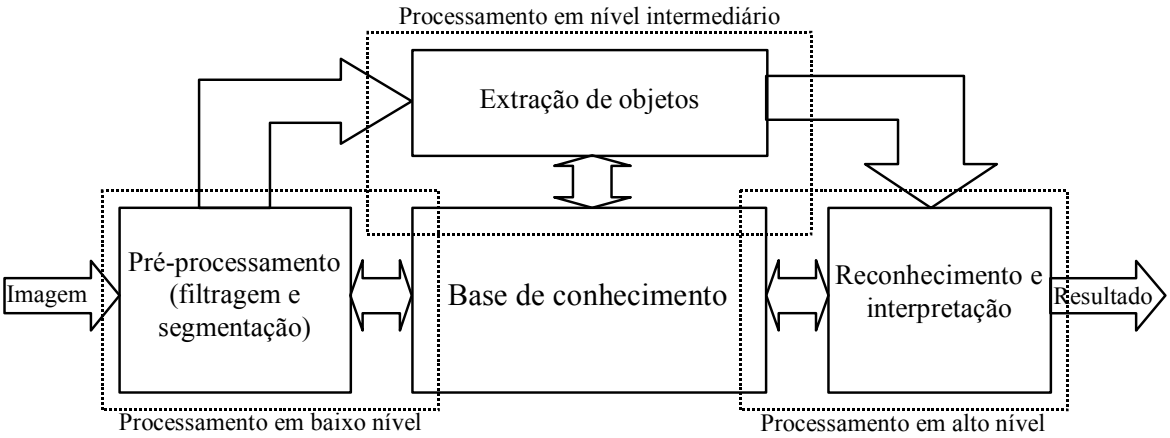


Figura 1.6 – Esquema de uma análise de imagens.

Uma análise de imagens, conforme o esquema da Figura 1.6, é dividida basicamente em três níveis de processamento: baixo, intermediário e alto. Todos os níveis são realizados de acordo com uma base de conhecimentos, onde estão armazenadas as informações sobre o objeto que se deseja extrair da imagem, que neste caso é a visualização do modelo ou padrão que se deseja acompanhar.

Em um processamento de baixo nível, são aplicadas técnicas de filtragem e segmentação às imagens, a fim de obter com mais facilidade os objetos no próximo nível. No processo de nível intermediário, são extraídos das imagens os objetos de interesse, que, no caso desta tese, são os segmentos de retas que podem fazer parte do modelo procurado. Finalmente, no último processo, é realizado um reconhecimento dos objetos extraídos, reconhecimento que é então interpretado. Desse modo, obtemos a partir do último processo os resultados desejados, os quais nesta tese são os parâmetros da câmera responsável pela visualização do modelo apresentada na imagem.

Faremos referência aos passos do algoritmo proposto de acordo com o fluxograma dado na Figura 1.7. Eles são detalhados nos próximos capítulos.

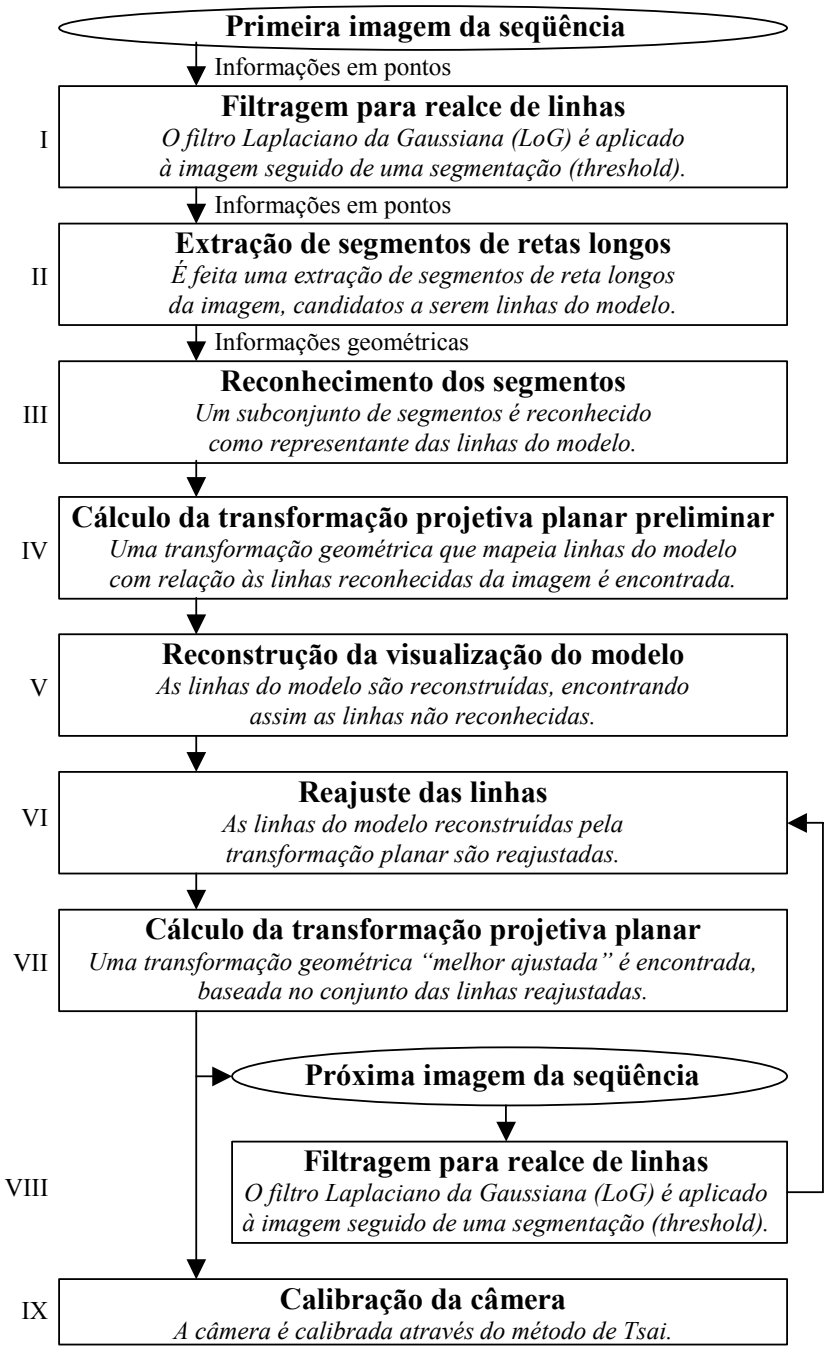


Figura 1.7 – Passos para acompanhamento de cena do algoritmo proposto.

Os passos I e VIII referem-se ao processamento de baixo nível da Figura 1.6, que realiza operações nos pontos das imagens, resultando em outras imagens. O passo II refere-se ao nível intermediário; são extraídos segmentos de retas candidatos a fazerem parte do

conjunto de segmentos da visualização do modelo. Já os demais passos referem-se ao processamento de alto nível.

1.4 Trabalhos Relacionados

Como foi visto, o algoritmo proposto para resolver o problema do acompanhamento de um modelo em uma seqüência de imagens possui diversos passos bem definidos e delimitados. Trata-se de passos de áreas de conhecimento separadas que apenas utilizam resultados obtidos em um passo anterior. A vantagem de um algoritmo com esta estrutura é que pode-se futuramente aprimorá-lo, alterando ou melhorando apenas um passo intermediário.

Pode-se dizer que o algoritmo proposto é uma coleção ordenada de outros algoritmos. Alguns algoritmos são bem conhecidos na literatura, como, por exemplo, o de filtragem da imagem para realçar linhas. Outros passos são contribuições desta tese para a área de visão computacional.

O modelo de câmera calibrada no final do segundo algoritmo proposto nesta tese é bem simples, do tipo "*pinhole*" [Shull99], utilizado por diversas bibliotecas gráficas, como a OpenGL [Woo+99]. Este modelo não considera, portanto, distorções causadas pelas lentes, apesar delas estarem presentes nas imagens. Um modelo de câmera mais real para computação gráfica é discutido em [Kolb+95].

No contexto geral do problema colocado (acompanhamento de modelo ou cenas), poder-se-ia questionar por que não são utilizados e aplicados algoritmos de acompanhamento de pontos ou linhas, descritos na literatura e conhecidos como métodos de *tracking*, como, por exemplo, os apresentados em [Clarke+96], [Blake+93], [Armstrong+95] e [Clarke+96a]. A resposta é simples: esses algoritmos de acompanhamento operam muito localmente nas imagens, isto é, realizam o acompanhamento de pontos ou linhas isoladamente, não levando em consideração o contexto da cena ou do modelo a ser acompanhado. Um modelo possui muito mais informações topológicas embutidas do que simplesmente retas ou pontos isolados, como adjacências de segmentos de retas. Acompanhar um modelo por inteiro exige um algoritmo robusto, que se preocupe em não perder a forma e topologia do modelo em questão, porém essa robustez não existe quando o algoritmo trabalha pontualmente ou de forma muito localizada.

No caso desta tese, mesmo com a câmera parada, podem existir objetos que se movem na seqüência, o que pode atrapalhar o acompanhamento de um modelo estático se for feito localmente. Os algoritmos descritos nos artigos mencionados acima relatam movimentos e acompanhamentos de pontos ou linhas apenas relativos ao quadro anterior. Em [Rahimi+01] é descrito um algoritmo de acompanhamento que estima localizações de objetos em um conjunto de quadros anteriores, além de novas posições de quadros futuros através de probabilidades. Em [Zeller+96], é apresentado um algoritmo para calibração automática de câmera dada uma seqüência de imagens utilizando equações de Kruppa e pontos que foram acompanhados ao longo da seqüência de imagens, os quais não formam obrigatoriamente nenhum modelo.

Existem dois tipos básicos de acompanhamento: visual, por intermédio de detectores e câmeras, e magnético, no qual geralmente são colados objetos no modelo e são feitas ligações entre esses objetos e o sistema. Em [State+96] é descrito um método de acompanhamento híbrido, que combina a precisão do método visual com a robustez do magnético, livrando-se de problemas como oclusão no método visual.

A idéia inicial desta tese é que o modelo seria apenas o conjunto das linhas do campo de futebol, mas com o desenvolvimento da tese, o modelo se tornou mais genérico. Para este propósito inicial, [Bebie+00] apresenta um método de reconstrução 3D baseado em vídeo, utilizando duas seqüências de vídeo sincronizadas de pontos de vistas diferentes, observando o mesmo lance do jogo. Nesse método, é realizado um acompanhamento de linhas no qual um usuário deve informar suas localizações no primeiro quadro de uma das seqüências. Porém, o objetivo desta tese é deixar de fora a interação com o usuário. No método de [Bebie+00], as posições dos jogadores e da bola também são fornecidas pelo usuário no primeiro quadro e, a partir do segundo, são acompanhadas automaticamente. Como são utilizadas duas câmeras, é possível calcular, em cada quadro, as posições 3D dos jogadores e da bola. Contudo, esta tese também não trata de encontrar de forma automática os jogadores e a bola.

Com o objetivo de acompanhar objetos em um domínio de futebol, no caso o americano, uma referência é [Intille+94]. Para o nosso futebol, tem-se como referência [Seo+97], [Kim+98] e [Kim+98a], e para *handball* [Perš+00]. Em [Perš+01] são analisados alguns erros que podem ser cometidos em um processo de acompanhamento automático de

jogadores.

Existem na literatura diversas referências para alguns passos do algoritmo proposto nesta tese. Um livro que aborda diversos desses passos é [Jain+95]. Uma boa referência para detecção de arestas é [Ziou+98], onde é apresentada uma visão geral das técnicas conhecidas, citando no final a detecção de linhas e junções. Nitzberg, em [Nitzberg+92], descreve um método de filtragem não linear para redução de ruídos realçando arestas, *corners* e junções tipo *T*, usando difusão anisotrópica baseado no método de suavização adaptativa em termos de difusão não linear, apresentado por Perona e Malik em [Perona+90], utilizando técnicas de filtragem em espaço de escala [Witkin83]. Clarke, em [Clarke+96], descreve um algoritmo de detecção de características lineares (segmentos de reta) através da divisão da imagem segundo uma grade retangular e em seguida um método de acompanhamento através da previsão de posição em um terceiro quadro dados dois anteriores. Clarke faz uma convolução de cada linha de cada retângulo da grade com uma derivada da Gaussiana ($0.0625\times[-3, -5, 0, 5, 3]$) para então estimar a componente de intensidade ao longo de cada linha. Quaisquer máximos locais de intensidade do gradiente maior que um determinado limiar de corte (tipicamente 30/256 do valor do ponto da imagem) são considerados arestas. Esse método, como é visto no artigo, pode apresentar como resultado, além de inúmeros segmentos curtos, segmentos que não existem na realidade na imagem. Porém, o objetivo principal do passo de detectar segmentos de retas no algoritmo proposto nesta tese é encontrar segmentos longos. Desejamos eliminar, além dos segmentos curtos, os segmentos falsos detectados por algum motivo – por exemplo, pela presença de pessoas ou regiões com indefinições, como é o caso de uma arquibancada de um estádio de futebol.

Em [Lee+95], é proposto um método rápido de detecção e ajuste de elementos lineares em imagens através da detecção de segmentos de retas e da realização de possíveis conexões com base em um funcional que depende dos ângulos formados entre as linhas existentes e as linhas reconstruídas pelas conexões. Os mesmos autores, em [Lee+97], aperfeiçoaram o método proposto para conexão de segmentos lineares utilizando o princípio de comprimento de descrição mínimo. Nesta tese, as conexões de segmentos de retas são feitas a partir de uma classificação das subimagens formadas pela grade e pelas retas suportes, com base em pequenas variações angulares. Um método de detecção de segmentos que se movem em uma seqüência de imagens geradas por apenas uma câmera também em movimento é descrito em

[Kasprzak+94].

Na área de reconhecimento de objetos e modelos 3D em uma imagem 2D, um artigo muito citado na literatura é [Lowe91], segundo o qual um objeto pode apresentar uma superfície curva arbitrária. Uma formulação projetiva do algoritmo proposto por Lowe é apresentada em [Araújo+98]. Em [Olson94] é descrito um método baseado em distribuição probabilística.

Apesar desta tese trabalhar inicialmente com imagens de um campo de futebol, que apresenta arcos de círculos, o algoritmo de detecção utilizado é apenas para segmentos de retas, pois já é o suficiente para que seja possível realizar o acompanhamento do campo e sua interpretação. Uma referência para detectar círculos, além de linhas, é [Vosselman+96]. Pernuš propõe em [Pernuš+94] um algoritmo de reconhecimento de objetos 2D baseado em análise de curvas utilizando técnicas de multi-resolução. Algumas limitações do reconhecimento baseado em modelos são discutidas em [Schweitzer+98].

Existem diversas técnicas de calibração de câmeras descritas na literatura. Com base em uma classificação feita por Tsai, em [Tsai86], podemos dividir estas técnicas em 4 categorias:

i. **Técnicas envolvendo apenas otimização não linear:** Estas técnicas possuem melhor precisão e facilidades de modelagem, mas necessitam de soluções iniciais boas para suas buscas (ou seja, não possuem autonomia) e exigem um alto esforço computacional. Exemplos de técnicas desta categoria são encontrados em [Faig75], [Abdel-Aziz+71], [Abdel-Aziz+74], [Karara79], [Brown71], [Sobel74], [Gennery79], [Malhotra71], [Wong75], [Okamoto81] e [Okamoto84].

ii. **Técnicas baseadas na transformação perspectiva usando resolução de equações lineares com possível segundo passo:** A principal vantagem das técnicas desta categoria é o fato de não utilizarem otimização não linear no primeiro passo (não necessitam de uma solução inicial). Uma desvantagem é a dificuldade de modelagem da câmera, levando o usuário a calibrar uma câmera mais simples, na qual geralmente não são levadas em consideração propriedades como a distorção da lente. Em muitas aplicações é necessário um segundo passo, que consiste na extração dos parâmetros (intrínsecos e extrínsecos) a partir da transformação perspectiva encontrada no primeiro passo. Neste caso, uma otimização não linear pode ser necessária. Exemplos destas técnicas são encontrados em [Yakimovsky+78],

[Sutherland74], [Strat84], [Ganapathy84], [Abdel-Aziz+71], [Abdel-Aziz+74], [Karara79], [Hall+82], [Faugeras93] e [Carvalho+98].

iii. **Métodos de dois planos:** São técnicas que exigem apenas a solução de um sistema de equações lineares mas, em compensação, o número de incógnitas é relativamente alto (no mínimo 24, 12 para cada plano). Estas técnicas calibram a câmera utilizando o sistema de coordenadas com base nos dois planos. Exemplos destas técnicas podem ser encontrados em [Martins+81] e [Isaguirre+85].

iv. **Técnicas geométricas:** São técnicas pouco difundidas e não necessitam de buscas não lineares. Os problemas estão relacionados com o fato de desconsiderarem distorções da lente, e a distância focal tem que ser conhecida juntamente com o fator de escala de imagem. Wang e Tsai, em [Wang+90], desenvolveram um método de calibração de parâmetros intrínsecos e extrínsecos baseado no conceito de linhas e pontos de fuga e usando um paralelepípedo regular como objeto de calibração. O modelo foi desenvolvido a partir da equação de colinearidade, encontrando-se expressões que relacionam os pontos de fuga com os parâmetros intrínsecos e extrínsecos. Os parâmetros intrínsecos calibrados são a distância focal e o centro da imagem. Não foram considerados a constante de distorção ótica nem o fator de escala em x. Em [Alvarez01], esta técnica é discutida e implementada em um sistema onde há uma cena arquitetônica urbana, na qual as construções presentes nas imagens nos levam a encontrar os três pontos de fuga. Como é apresentado nesta tese, o padrão visualizado a ser acompanhado é planar, portanto esta técnica não pôde ser utilizada pois existem apenas dois pontos de fuga.

Outras referências que apresentam métodos de calibração de câmeras para diversas situações são: [Svedberg+99], [Li+94], [Li+95], [Li+96], [Spiess+96], [Bellon+90] e [Mendonça+99]. Tommaselli e Tozzi, em [Tommaselli+91], fazem um levantamento de diversas técnicas de calibração de câmeras dentro das categorias enumeradas acima. Em [Rousso+96] é apresentado um método para recuperar rotações realizadas em um câmera a partir de três quadros consecutivos de uma seqüência.

Realizando-se um processo de calibração de câmeras, pode ser possível recuperar algumas informações 3D a partir de uma cena. Um algoritmo para tal técnica é apresentado em [Carvalho+98], onde uma cena de uma partida de futebol é reconstruída, incluindo jogadores, juiz e bola. Outra referência para esta técnica é [Debevec+96], onde é modelado

um cenário urbano.

Seguindo a linha de calibração de câmeras para cenas estáticas, é natural explorar estas técnicas para uma seqüência de cenas. Um algoritmo do tipo força bruta com esta finalidade consiste em repetir o processo de calibração de câmera para cada cena. Entre as desvantagens desta técnica simples estão o alto esforço computacional e muitas vezes um alto grau de intervenção do usuário, pois ela não leva em conta a coerência de quadros subseqüentes. Os algoritmos que trabalham com uma seqüência de cenas, realizando as devidas calibrações de câmeras para as respectivas imagens, são geralmente denominados algoritmos de auto-calibração de câmeras.

Zeller e Faugeras apresentam, em [Zeller+96], um algoritmo de auto-calibração de câmeras a partir de uma seqüência de imagens baseado nas equações de Kruppa. O algoritmo se baseia em técnicas de rastreamento ou acompanhamento de pontos, também chamadas técnicas de *tracking* de pontos.

1.5 Organização da Tese

Nos capítulos que se seguem, são apresentados em detalhes os métodos utilizados nos dois algoritmos propostos. No Capítulo 2, diversos filtros comuns presentes na literatura são analisados com o objetivo de facilitar a extração de segmentos da imagem. Um dos meios de fazer isto é eliminar ruídos presentes na imagem. Neste mesmo capítulo, são apresentados métodos de aplicação de segmentação por limiar, conhecidos como técnicas de *threshold*, disponíveis na literatura com o objetivo de segmentar uma imagem e extrair pontos candidatos a estarem sobre uma linha. Todos os métodos apresentados no Capítulo 2 já são conhecidos na área de processamento de imagens, mas são aplicados às nossas imagens de interesse e observados seus resultados, a fim de verificar sua utilidade para o nosso caso.

No Capítulo 3, são apresentados dois algoritmos de extração de segmentos de retas. O primeiro, chamado Transformada de Hough, é bem difundido na literatura e, por ser clássico, foi feita a tentativa de utilizá-lo. Porém, como é visto, ele não apresenta bons resultados quando aplicado a uma imagem real. Assim, ainda neste capítulo, é apresentado um novo algoritmo, proposto nesta tese, baseado na divisão da imagem em células segundo uma grade regular.

O Capítulo 4 trata de reconhecer as linhas extraídas da imagem baseando-se em

modelos. O primeiro algoritmo apresentado é uma adaptação da Transformada de Hough e são descritas suas características. Em seguida é apresentado um algoritmo de reconhecimento baseado em interpretações e restrições. De início, as interpretações e restrições são relacionadas ao modelo que motivou esta tese, ou seja, linhas do campo de futebol, e a partir daí são generalizadas para o caso de modelos compostos por segmentos de retas perpendiculares ou ortogonais entre si.

No Capítulo 5 é apresentado o modelo de câmera utilizada nesta tese, enumerando cada um de seus parâmetros. Ainda neste capítulo, são descritos dois métodos para encontrar as homografias usadas no algoritmo, transformações projetivas planares, e dois algoritmos de calibração de câmeras, sendo um clássico da literatura, o Método de Tsai, adaptado ao nosso modelo simplificado de câmera.

Para terminar a apresentação dos métodos utilizados, o Capítulo 6, dedicado ao acompanhamento de cenas, apresenta uma forma de encadear todos os passos do algoritmo ilustrado na Figura 1.7, cujos métodos foram apresentados nos capítulos anteriores. São utilizados os métodos apresentados nos capítulos anteriores como ferramentas e meios combinados em um único algoritmo para calibração automática de câmera estimando inicialmente a localização do modelo na imagem.

O Capítulo 7 descreve as implementações feitas empregando o segundo algoritmo proposto em situações diversas: usando imagens reais de uma partida de futebol extraídas de uma transmissão de televisão; utilizando uma câmera 8mm capturando imagens de um campo de futebol de botão e de um outro padrão; e outra usando uma "*webcam*", que apresenta alta distorção de lente, que o algoritmo não trata. Para testar a robustez do algoritmo, nos casos em que as imagens não são de uma partida real de futebol, partes do modelo são escondidas. Em seguida, são enumeradas algumas aplicações em que podem ser utilizados os algoritmos propostos.

O Capítulo 8 traça conclusões sobre o algoritmo proposto, obtidas a partir de resultados de implementação. No final, são enumeradas algumas propostas de trabalhos futuros.

Finalmente, é listada a bibliografia utilizada ao longo do desenvolvimento desta tese, juntamente com as demais referências.

Todos os capítulos desta tese apresentam no final conclusões dos métodos descritos.

Capítulo 2

2 Pré-Processamento: Filtragem e Segmentação

Este capítulo aborda os passos I e VII do algoritmo para acompanhamento de cena proposto nesta tese, ilustrado na Figura 1.7. Eles procuram determinar os pontos que se localizam sobre os segmentos de retas que representam as linhas em cada imagem da seqüência de vídeo. Para estabelecer a nomenclatura, apresentaremos inicialmente uma breve formalização matemática e em seguida discutiremos alguns filtros e processos de segmentação de imagens disponíveis na literatura que consideramos adequados para o problema em questão.

Entre as imagens para ilustrar os métodos escolhidos está uma mais usada nesta tese, capturada de uma transmissão de TV, mostrada na Figura 2.1. Esta imagem é uma visualização da região próxima a grande área de um campo de futebol, apresentando diversos problemas de transmissão, captura e complexidade de cena. Os dois primeiros problemas produzem interferências e falta de nitidez, enquanto o último refere-se à textura do gramado e à presença de outros objetos, tais como jogadores, arquibancadas e painéis de propaganda ao redor do campo. A interferência, falta de nitidez e textura dificultam a identificação dos pontos das linhas. A textura e os outros objetos tendem a fazer com que os métodos de detecção encontrem linhas não desejadas, e os jogadores podem esconder parte das linhas procuradas. Um complicador adicional a esses problemas é a presença de sombras em partes

da imagem. Outras imagens são sintéticas ou também capturadas, mostrando outras regiões próximas a grande área de um campo de futebol.

Figura 2.1 – Imagem capturada de uma transmissão de televisão.

2.1 Imagens de Vídeo

O objeto de trabalho nesta tese, onde são aplicadas as técnicas do algoritmo de acompanhamento de cena proposto, é um vídeo. Um vídeo é uma seqüência S de imagens I_t, onde I_t indica a imagem no tempo t do vídeo. É comum chamar uma imagem do vídeo de quadro (ou *frame*) e uma propriedade importante do vídeo refere-se à sua taxa de amostragem, medida em quadros por segundo (*fps*). Em uma animação de boa qualidade, esta taxa deve estar próxima de 30 quadros por segundo. Essa é a taxa utilizada pela televisão, portanto os equipamentos de captura de vídeo procuram capturar também a esta taxa. Em alguns casos, a taxa é variável ao longo do vídeo.

Uma imagem I pode ser descrita como sendo uma função

$$I : D \subset \Re^2 \rightarrow \Re^n \tag{2.1}$$

onde D é o domínio da imagem, que em geral é um retângulo limitado e discreto, e n é um número inteiro maior que zero, que equivale ao número de escalares utilizados para representar as informações de cor para um ponto qualquer da imagem.

Na prática, cada um destes escalares é quantizado para um certo número de bits. A situação mais freqüente é se utilizar um byte para cada escalar, que é então representado por um número de 0 a 255.

Nesta tese, a imagem correspondente ao plano de projeção ou do filme da câmera a ser calibrada no último passo do algoritmo proposto é referenciada da forma mostrada na Figura 2.2. As resoluções horizontal e vertical são denotadas por w e h, respectivamente, e as dimensões por $w\Delta u$ e $h\Delta v$. Os incrementos Δu e Δv são tomados como iguais a 1. Conceitualmente, cada ponto da imagem (*pixel*) é localizado pelo par ordenado (u, v), e sua informação de cor é definida por i_{uv}. Na memória principal e na captura de vídeo, esta informação é armazenada através de um vetor, onde o ponto de coordenadas u e v é armazenado na posição $uw + v$.

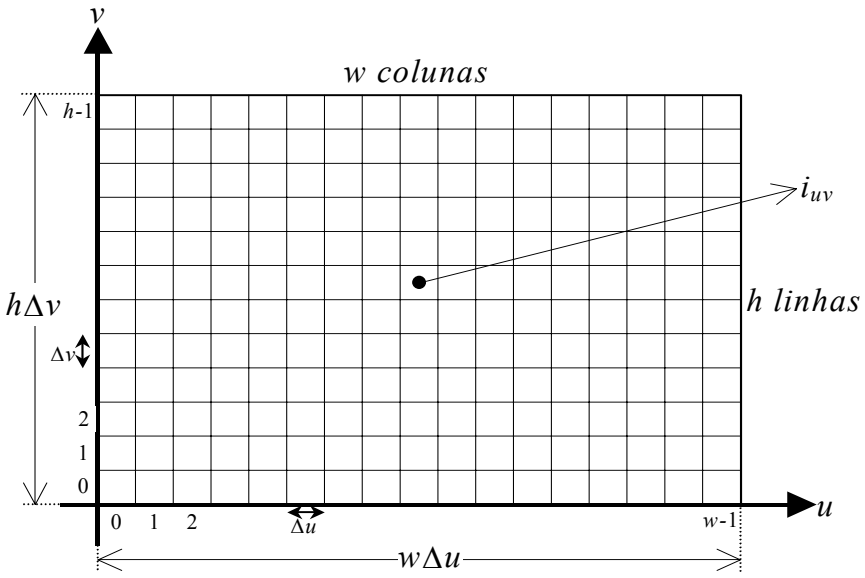


Figura 2.2 – Formato de uma imagem.

Cada equipamento de captura de vídeo pode possuir uma forma particular de descrição do vídeo e de suas imagens. Um vídeo pode conter compressões e as cores das imagens podem ser dadas pela decomposição nas cores primárias vermelho (R), verde (G) e azul (B), por sua luminância e cromaticidade ou outra forma ([Tekalp95] e [Symes98]).

Não trataremos, nesta tese, de vídeos comprimidos, portanto o vetor que descreve uma imagem de cores primárias de um vídeo capturado tem tamanho três vezes maior que o tamanho da imagem, sendo a imagem varrida linha a linha e a cor de um ponto (u, v) dada nas

posições 3.($v + u.w$), 3.($v + u.w$)+1 e 3.($v + u.w$)+2. Convém observar que as imagens de um vídeo capturado têm seus canais de cores invertidos com relação ao convencional, isto é, a cor de um ponto é dada pela tripla *BGR* em vez de *RGB*.

Alguns filtros e transformações apresentados neste capítulo têm como entrada ou saída imagens monocromáticas. Nesta tese, um filtro é tratado como uma transformação onde é aplicada uma máscara segundo a operação de convolução. As imagens monocromáticas são caracterizadas por terem o valor de n igual a 1 na equação (2.1) ([Gonzales+93]) e podem ser representadas por tons de cinza. Outro tipo de imagem monocromática é a binária, que apresenta apenas dois valores (geralmente tomados como 0 e 1) para seus pontos. Essas imagens são geralmente usadas como resultado de um processo de segmentação. Quando a imagem resultante é binária, estamos interessados apenas em separar pontos da imagem que satisfazem ou não uma certa propriedade. Usamos também imagens em tons de cinza, nas quais, além de separar tais pontos, podemos incorporar outras propriedades importantes, como, por exemplo, um grau de certeza da presença de um segmento de reta em um determinado ponto.

Tratamos aqui duas maneiras de transformar uma imagem colorida em monocromática do tipo tons de cinza. A primeira consiste em descartar dois de seus canais de cor, geralmente o vermelho e o azul. A segunda é obtida pelo cálculo da luminância, dado por:

$$\boldsymbol{L} = 0.299\boldsymbol{R} + 0.587\boldsymbol{G} + 0.114\boldsymbol{B} \tag{2.2}$$

onde $\boldsymbol{R}$, $\boldsymbol{G}$ e $\boldsymbol{B}$ definem a cor de um ponto.

O descarte de dois canais de cor é mais rápido, porém a utilização do cálculo da luminância pode produzir resultados mais satisfatórios no final do algoritmo. Isto ocorre porque, ao descartar dois canais de cor, podemos perder informações importantes que em muitos casos são incorporadas no calculo da luminância. Na Figura 2.3 é ilustrada a transformação em tons de cinza através do cálculo da luminância aplicado à Figura 2.1.

Na segmentação de linhas com a espessura das normalmente encontradas nas imagens de jogos de futebol é interessante considerar (como é mostrado a seguir) o negativo de uma imagem. No caso de uma imagem em tons de cinza quantizada em um byte (valores de 0 a 255), seu negativo é dado por:

$$\boldsymbol{N} = 255\text{-}\boldsymbol{I} \tag{2.3}$$

onde ***I*** é a intensidade de cor de um ponto da imagem. Esta equação aplica-se a imagens em tons de cinza. Para imagens coloridas, deve-se aplicar esta equação para cada canal de cor.

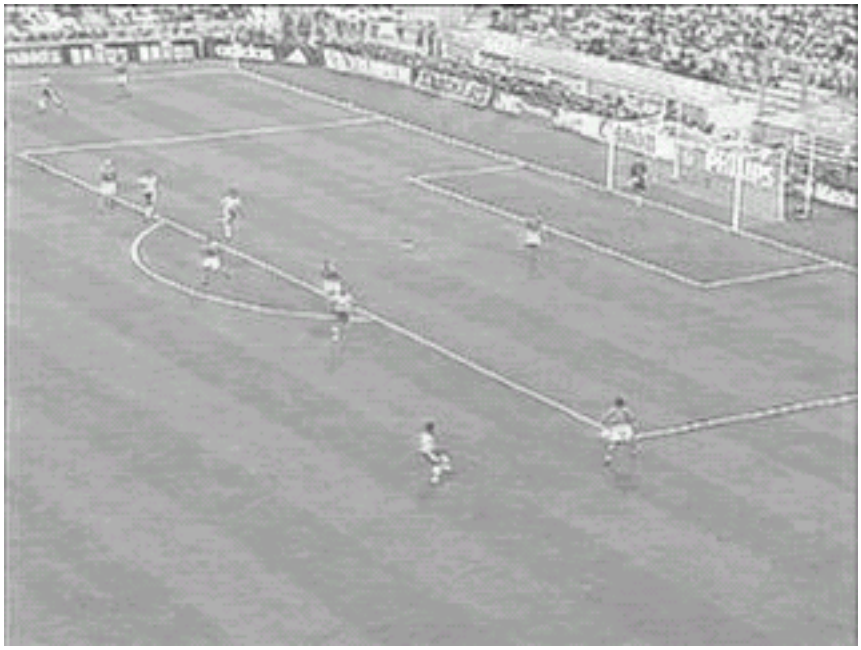


Figura 2.3 – Imagem em tons de cinza baseada na luminância aplicada à Figura 2.1.

A Figura 2.4 ilustra a transformação negativa aplicada à Figura 2.3.

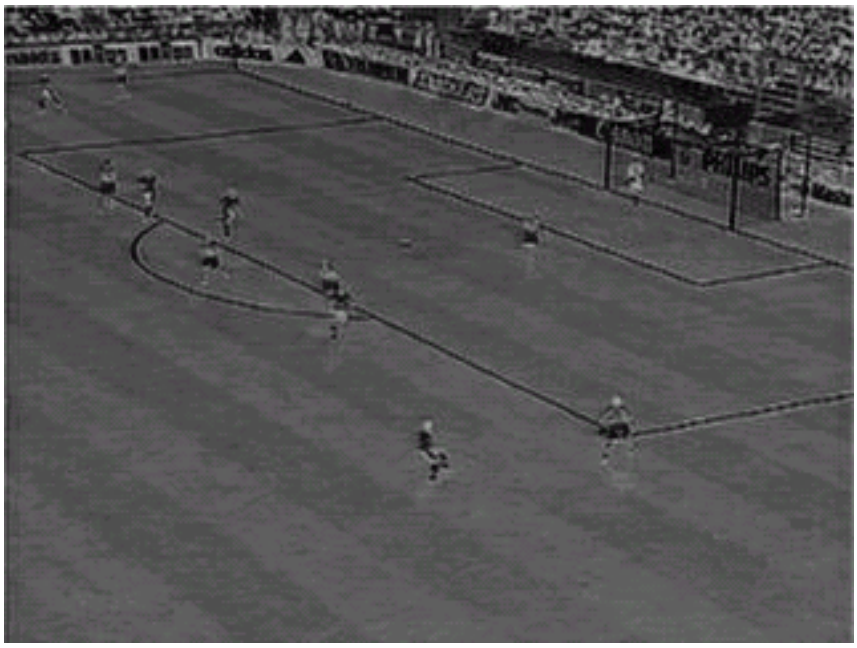


Figura 2.4 – Negativo da Figura 2.3.

2.2 Detecção de Segmentos de Retas Simples

Como foi dito na introdução do capítulo, estamos interessados em determinar um conjunto de pontos da imagem que se localizem sobre segmentos de reta. Para este objetivo, aplicamos filtros que detectam pontos isoladamente candidatos a fazerem parte desse conjunto, através de realce de linhas. A seguir é aplicada uma técnica para descartar alguns pontos pouco prováveis do conjunto através do emprego de um limiar de corte (algoritmo de *threshold*).

2.2.1 Filtragem de Imagens – Realce de Linhas

Com o objetivo de detectar pontos candidatos a estarem sobre um segmento de reta presente na imagem, é empregada nesta tese a filtragem laplaciana definida a partir do operador matemático laplaciano ∇, bem difundida na literatura ([Gonzales+93]) e muito utilizada em diversas aplicações para este fim, classificada como filtro passa-altas (freqüências). A matriz 3x3 que representa o núcleo da convolução da filtragem laplaciana em um domínio discreto é dada por:

0	1	0
1	-4	1
0	1	0

Este filtro aplica-se a imagens monocromáticas em tons de cinza. Para exemplificar a aplicação do filtro laplaciano, é usada a imagem ilustrada na Figura 2.3 e o resultado é apresentado na Figura 2.5.

Como podemos observar, existem muitos pontos com intensidade alta na Figura 2.5 que não se encontram sobre nenhum segmento de reta. Em muitos casos, isto se deve à presença de interferências e ruídos, pois trata-se de uma imagem proveniente de uma transmissão de televisão, ou então da textura do gramado. Para suavizar este problema, realiza-se uma filtragem do tipo passa-baixa, eliminando altas freqüências. O filtro mais comum (e utilizado nesta tese) é o gaussiano [Babaud+86]. O núcleo da convolução do filtro gaussiano utilizado nesta tese é dado pela seguinte matriz, multiplicado pelo fator $\frac{1}{16}$:

1	2	1
2	4	2
1	2	1

Figura 2.5 – Resultado da filtragem laplaciana aplicada à Figura 2.3.

A Figura 2.6 ilustra a aplicação da filtragem gaussiana à imagem da Figura 2.3.

Figura 2.6 – Imagem resultante da filtragem gaussiana aplicada à Figura 2.3.

Aplicando o filtro laplaciano à imagem da Figura 2.6, obtemos a imagem ilustrada na Figura 2.7. Observe que agora as linhas presentes na imagem ficaram mais nítidas.

A composição dos dois filtros acima é conhecida como filtragem laplaciana da gaussiana, também chamada na literatura de *LoG* (do inglês *Laplacian of Gaussian*) ([Gonzales+93]).

Figura 2.7 – Resultado da filtragem laplaciana aplicada à imagem resultante de uma filtragem gaussiana.

Observe que as linhas que representam as marcações do campo de futebol, segmentos de retas que desejamos detectar, apresentam-se na forma de duas linhas paralelas. Isto é resultado do fato de que o filtro laplaciano encontra fronteiras de regiões, onde localizam-se as altas freqüências – ou seja, foram detectadas regiões de fronteira; por exemplo, no caso das linhas do campo, os limites entre as linhas e o gramado. Isto irá dificultar a extração das linhas verdadeiras do campo no passo posterior do algoritmo proposto.

No entanto, se for feita uma outra composição, agora incluindo a transformação negativa antes do filtro *LoG*, obtém-se a imagem ilustrada na Figura 2.8.

Pode-se notar que a principal diferença entre a Figura 2.7 e a Figura 2.8 é que a primeira apresenta linhas paralelas onde a segunda apresenta linhas simples. Nesta tese, o resultado da Figura 2.8 é o desejado, pois detectou melhor os pontos localizados que estão sobre as linhas do campo de futebol.

Figura 2.8 – Resultado da filtragem *LoG* aplicada a negativa da Figura 2.3.

A diferença entre as detecções das linhas presentes nestas duas figuras pode ser entendida observando o gráfico das intensidades ao longo de um corte perpendicular a uma linha do campo, como mostrado na Figura 2.9. O gráfico obtido é similar ao gráfico da função

$$y = \frac{255}{1 + x^2} \tag{2.4}$$

ilustrado na Figura 2.9.

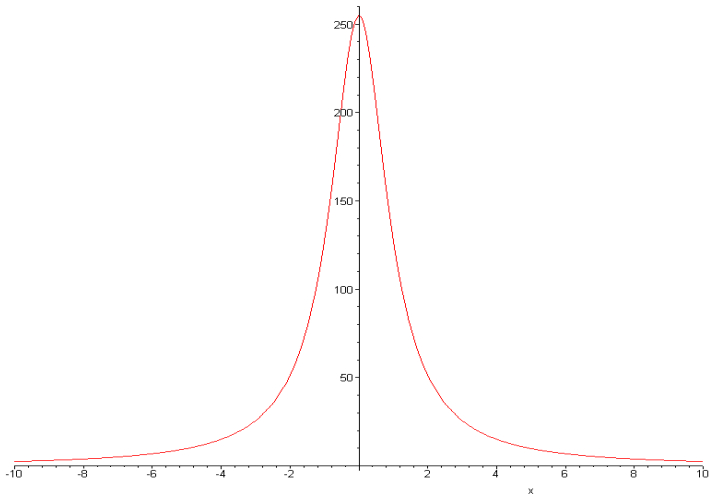


Figura 2.9 – Esquema de um corte uma linha de um campo de futebol.

O gráfico do laplaciano aplicado a esta função é ilustrado na Figura 2.10.

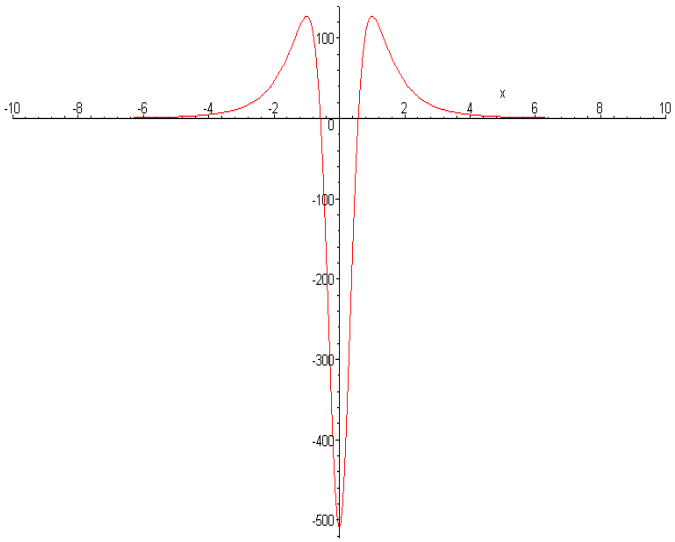


Figura 2.10 – Laplaciano do gráfico da Figura 2.9.

Podemos notar neste gráfico a existência de dois picos. Eles representam as linhas paralelas surgidas na filtragem ilustrada na Figura 2.7.

Mas, se aplicarmos a transformação negativa dada pela equação (2.3) em (2.4), obtemos a expressão (2.5)

$$y = 255 - \frac{255}{1+x^2} = \frac{255x^2}{1+x^2} \tag{2.5}$$

Seu gráfico é ilustrado na Figura 2.11,

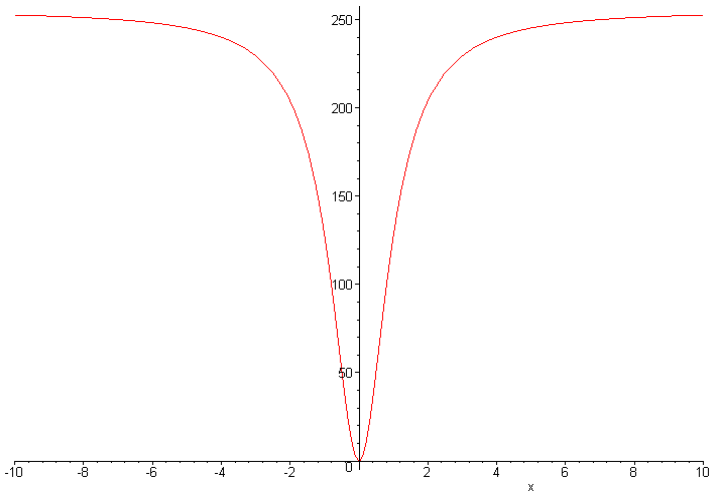


Figura 2.11 – Gráfico da transformação negativa aplicada à Figura 2.10.

e o laplaciano desta função terá o gráfico ilustrado na Figura 2.12.

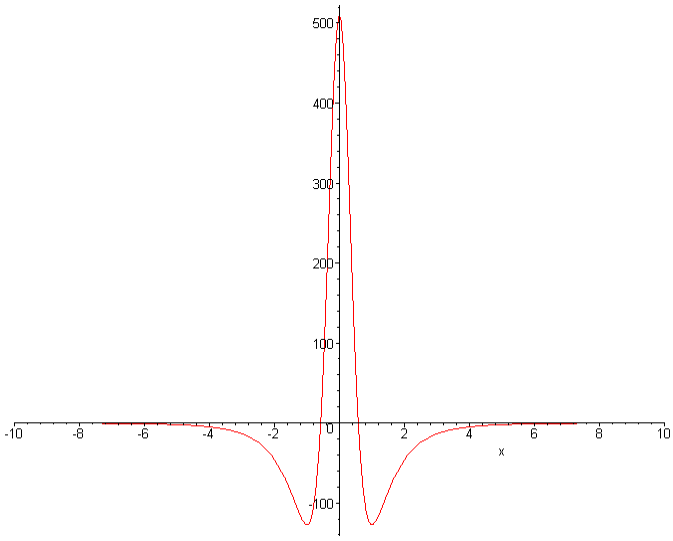


Figura 2.12 – Laplaciano do negativo.

Diferente do gráfico da Figura 2.10, este gráfico apresenta apenas um pico, representando uma linha simples. Com isto, os pontos que estão localizados sobre linhas do campo ficam mais nítidos na imagem ilustrada na Figura 2.8.

2.2.2 Segmentação de Imagens

Nesta etapa, estamos interessados em obter, a partir da imagem original, uma imagem resultante com a propriedade de que cada ponto indique a possível presença de uma reta que passa por ele. Isto é, são excluídos da imagem original, através da atribuição do valor 0 (zero) na imagem resultante, os pontos que não estão sobre nenhum segmento de reta. Para os pontos não excluídos, são atribuídos valores, interpretados como o peso a eles associado, indicando um grau de certeza de passagem de um segmento de reta, utilizados no passo de extração de segmentos de reta da imagem.

Em geral, a segmentação de imagens se faz através de duas técnicas: aplicação de um limiar ou valor de corte, conhecido na literatura como técnica de *thresholding*, e coerência. A primeira técnica é abordada nesta tese. A segunda não é discutida aqui para realizar segmentação de imagens, pois o objetivo da tese é obter resultados em tempo real, o que não seria possível com a utilização da técnica de coerência. Contudo, uma técnica de coerência é

utilizada pelo algoritmo proposto nesta tese para extração de segmentos de reta da imagem, utilizado no passo II do algoritmo ilustrado na Figura 1.7, que é descrito no Capítulo 3.

A técnica de segmentação por limiar é uma das mais importantes ferramentas utilizadas na segmentação de imagens. O objetivo deste tipo de filtro é separar uma parte da imagem das demais através de um determinado valor de corte e uma função de avaliação, isto é, a imagem é avaliada ponto a ponto e cada ponto é classificado de acordo com o valor retornado pela função de avaliação.

Muitas vezes, necessita-se avaliar o comportamento de um ponto com seus vizinhos, através da aplicação de outros filtros, para depois então poder aplicar a técnica de segmentação. Isto ocorre, por exemplo, quando estamos interessados em detectar descontinuidades. Os três tipos mais importantes de descontinuidade em uma imagem digital são pontos, linhas e arestas. Arestas aqui significam fronteiras entre duas regiões. Na prática, o método mais comum de detectar descontinuidades se dá através da aplicação de uma máscara em cada ponto da imagem e da análise de seu resultado via segmentação por limiar.

Nesta tese, o filtro utilizado é a composição da transformação negativa com o filtro *LoG*, descrito na Seção 2.2.1. O valor de cada ponto da imagem resultante da aplicação dessa composição de filtros é comparado com um determinado limiar de corte: caso o valor seja maior que o limiar, então o ponto é de interesse do algoritmo; caso contrário, o ponto é descartado. Vale lembrar que estamos interessados em obter pontos candidatos a estarem sobre um segmento de reta contido na imagem.

Existem diversos métodos para se obter os valores de corte com base em propriedades da imagem, sendo um dos mais conhecidos o método de Otsu [Otsu79]. Ele se baseia em encontrar duas modas no histograma de cores da imagem e em seguida calcular, utilizando métodos estatísticos, um valor que separe estas duas modas, estabelecendo uma fronteira entras elas. Este é o valor utilizado como limiar de corte para o método de segmentação.

Empregando o método de Otsu para determinar automaticamente o limiar de corte na Figura 2.3, obtemos o resultado apresentado na Figura 2.13. Pode-se observar a presença de muitos pontos que ainda não estão sobre nenhum segmento de reta. Isto se deve ao fato de que o método de Otsu não produz bons resultados quando uma imagem não apresenta duas modas bem definidas, como é o caso da imagem utilizada, cujo histograma é mostrado na Figura 2.14 juntamente com o valor de corte.

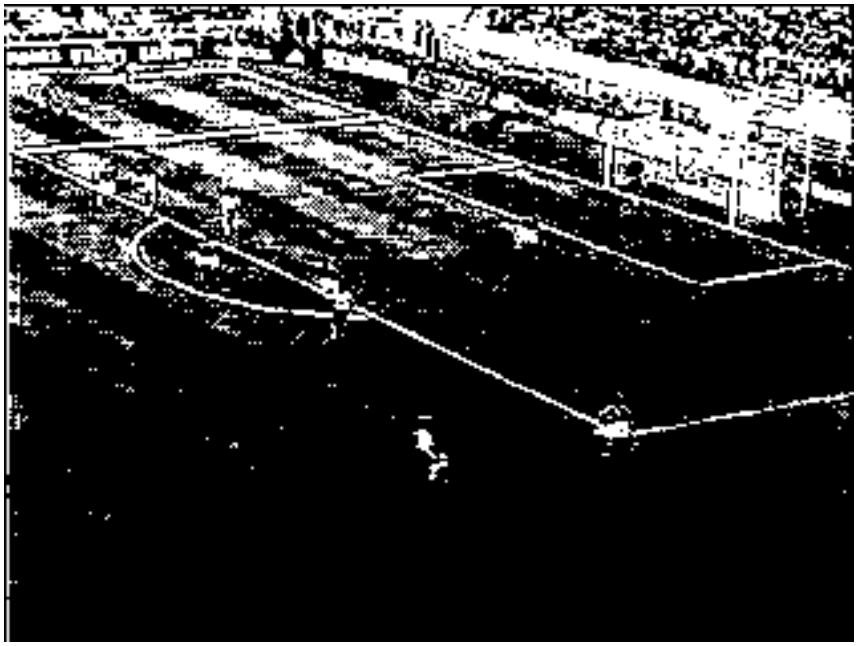


Figura 2.13 – Segmentação da Figura 2.3 com o método de Otsu.

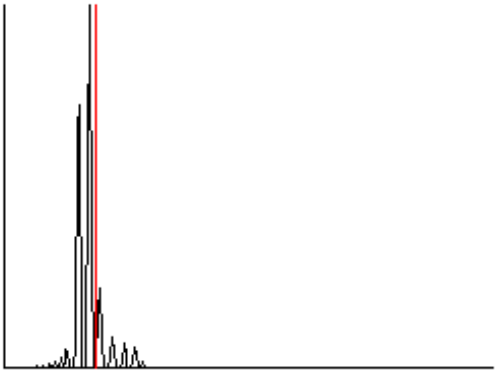


Figura 2.14 – Histograma de cores da Figura 2.3 com o valor de corte (traço vermelho) determinado pelo método de Otsu.

Mesmo utilizando o método de Otsu para a Figura 2.8, que é o resultado da transformação negativa seguida do filtro *LoG* aplicado à Figura 2.3, o resultado ainda apresenta pontos com intensidades altas, conforme visto na Figura 2.15. O histograma de cores da Figura 2.8 é ilustrado na Figura 2.16 juntamente com o valor de corte.

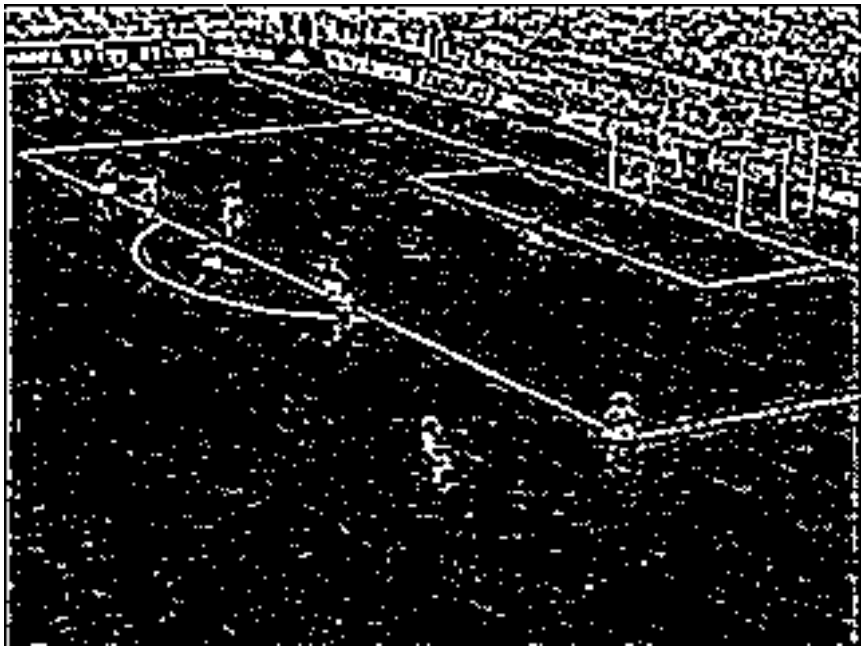


Figura 2.15 –Segmentação por limiar aplicado à Figura 2.8 utilizando o método de Otsu.

Figura 2.16 – Histograma de cores da Figura 2.8 com o valor de corte (traço vermelho) determinado pelo método de Otsu.

Para a finalidade deste trabalho, obtêm-se resultados melhores sem a utilização do método de Otsu, como ilustra a Figura 2.17. Esta imagem é binária e resulta de uma aplicação da segmentação por limiar na Figura 2.8 com valor de corte igual a 20.

Note que a maioria das linhas do campo na figura acima foi detectada, embora outros objetos como jogadores, traves e arquibancada também tenham sido detectados, sendo portanto indesejados. A eliminação desses pontos indesejados é vista no próximo capítulo, juntamente com a técnica de extração dos segmentos.

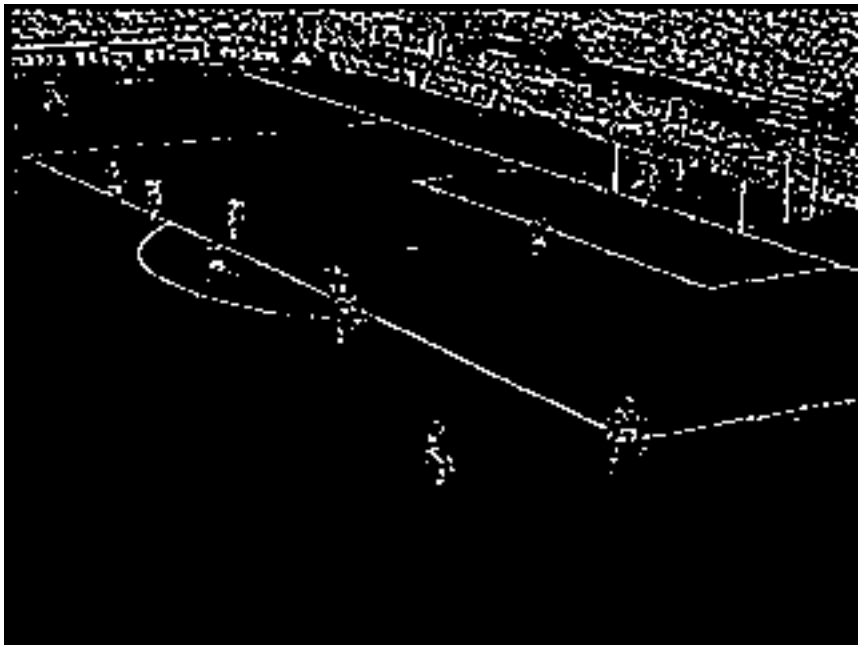


Figura 2.17 – Segmentação aplicada à Figura 2.8 com limiar de corte igual a 20.

Existem outros exemplos onde o resultado da utilização do método de Otsu pode ser ainda pior. Este é o caso quando ele é usado para a Figura 2.18, obtendo o resultado mostrado na Figura 2.20. Este resultado se dá porque o histograma, ilustrado na Figura 2.19, apresenta apenas uma moda e muito bem definida.

Figura 2.18 – Exemplo onde o método de Otsu falha.

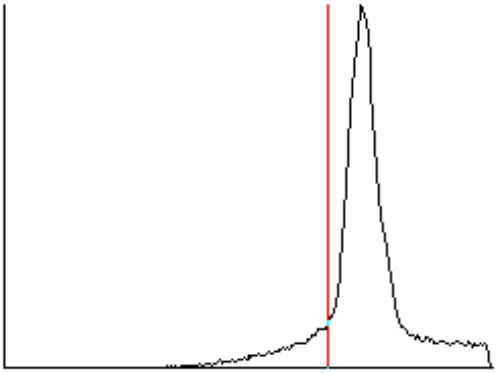


Figura 2.19 – Histograma de cores da Figura 2.18 com o valor de corte determinado pelo método de Otsu.

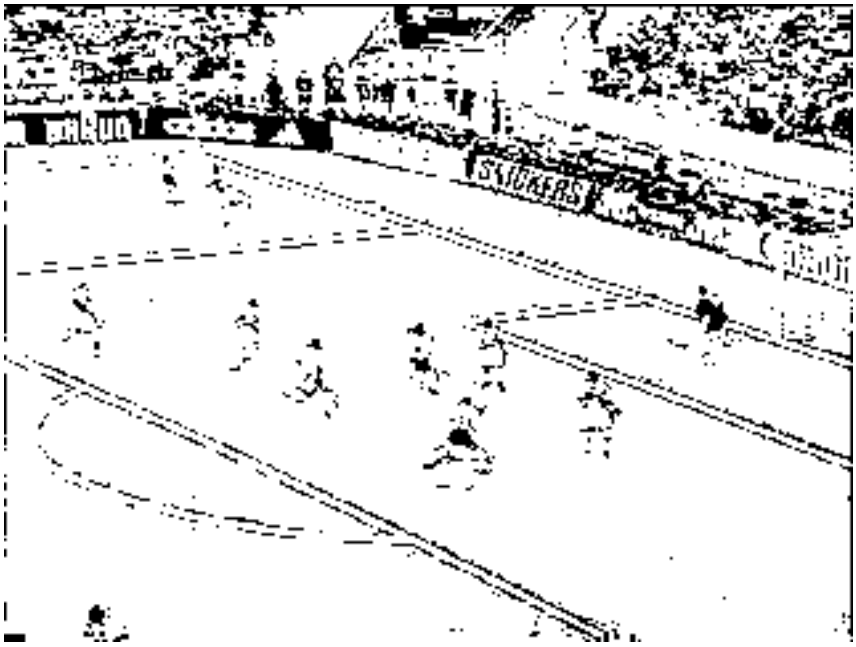


Figura 2.20 – Resultado da segmentação com o valor de corte determinado pelo método de Otsu para a Figura 2.18.

Aplicando a composição da transformação negativa com *LoG* à Figura 2.18, temos a Figura 2.21.

Figura 2.21 – Transformação negativa com filtro *LoG* aplicada à Figura 2.18.

Utilizando o método de Otsu para determinar o limiar de corte a ser utilizado para segmentar a Figura 2.21, cujo histograma de cores está ilustrado na Figura 2.22, onde a linha vermelha indica o limiar de corte, obtemos o resultado ilustrado na Figura 2.23. Observamos novamente que existem muitos pontos com intensidades altas que não estão sobre nenhuma linha do campo da Figura 2.18.

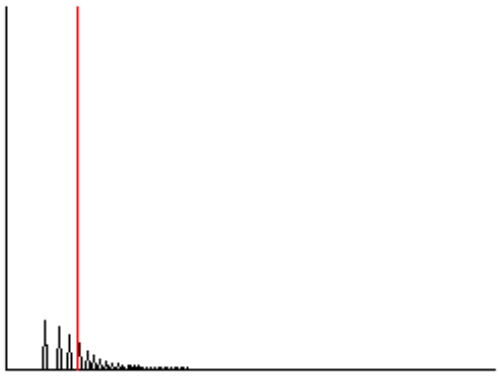


Figura 2.22 – Histograma de cores da Figura 2.21.

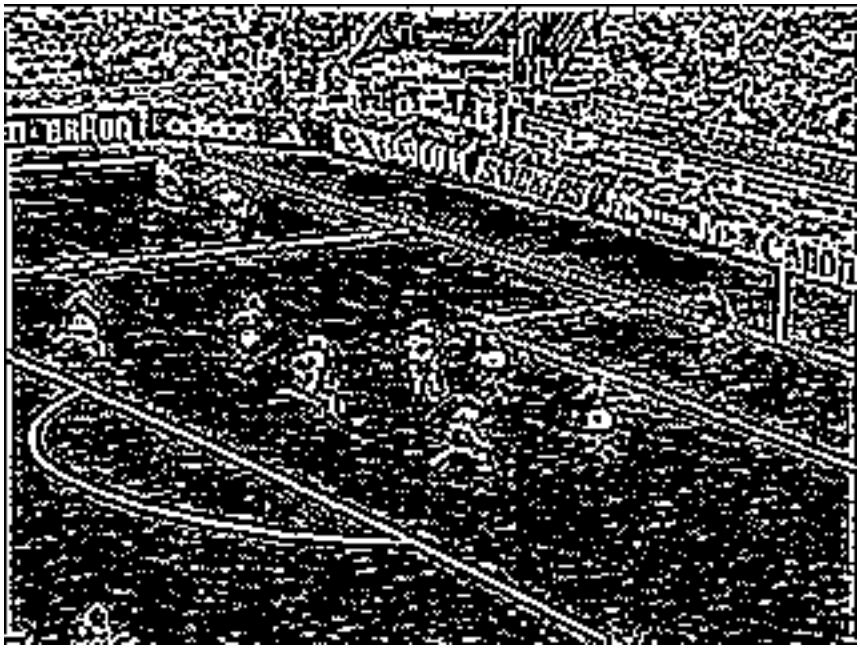


Figura 2.23 – Resultado da utilização do método de Otsu na Figura 2.21.

Determinado manualmente o limiar de corte igual a 16, temos o resultado da segmentação apresentado na Figura 2.24.

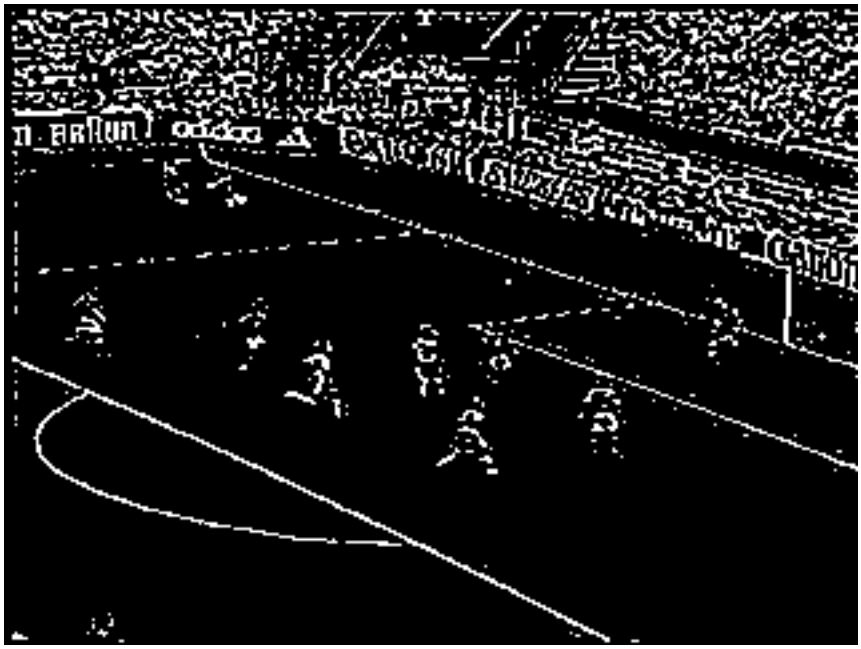


Figura 2.24 – Resultado obtido com a aplicação prévia da transformação negativa com o filtro *LoG* e determinação manual do limiar de corte.

Cogitou-se utilizar uma heurística para obter automaticamente os valores de corte, baseada em funções de acumulação de intensidade, na qual os valores de corte são dados pelos pontos onde as funções apresentam a maior derivada. Obteve-se melhores resultados, no entanto, com uma outra heurística, baseada no número de segmentos extraídos, descrita na Seção 6.4.

2.3 Conclusão

Neste capítulo foram apresentados transformações e filtros com o objetivo de realçar pontos de interesse em imagens, que no caso desta tese são pontos localizados sobre algum segmento de reta. Foram apresentados também processos de segmentação para extrair os pontos de interesse da imagem e tentar automatizar a determinação dos parâmetros para esta finalidade.

Em relação a transformação e filtragem, foi visto que, na maioria das vezes, é necessário aplicar uma composição de diversos filtros e transformadas. Este é o caso quando as imagens são provenientes de uma transmissão de televisão, pois elas apresentam, entre outros problemas, interferência de sinais e ruídos, prejudicando o processo de localização dos pontos de interesse. Portanto, é necessário realizar uma filtragem para eliminar estes ruídos através de um certo borramento na imagem. Por outro lado, quando fazemos este borramento, poderemos estar perdendo informações importantes que não são ruídos mas que estão relacionados com a nitidez da imagem. Este é outro problema com estes tipos de imagens – em virtude da captura de imagens através da câmera e da reconstrução feita pelo aparelho de televisão, a nitidez da imagem reproduzida pode, às vezes, ser baixa. Outros problemas com imagens de televisão estão relacionados com o gamute de cores e as bordas, que apresentam imperfeições mas que são eliminados pelos aparelhos de televisão. As aplicações de determinados filtros atenuam estes problemas, apresentando resultados mais satisfatórios, como é o exemplo da aplicação do filtro *LoG*. Entretanto, estes problemas não são apenas de responsabilidade da captura e transmissão. Como pudemos observar com imagens reais de uma partida de futebol, alguns fatores prejudicavam o realce dos pontos de interesse, entre os quais podem ser citados:

- Textura do gramado: muitas vezes o gramado apresenta falhas e diversas tonalidades de cores, além de, em alguns gramados, a direção do estádio colocar

imagens em tons de verde, como, por exemplo, desenho de xadrez e até a bandeira do país.

- Linhas mal definidas: em algumas imagens podemos notar que as linhas do campo estavam com pinturas fracas ou então a iluminação ambiente atrapalhava a sua visualização.
- Presença da arquibancada, propagandas ao redor do campo, traves e rede do gol e outros itens presentes no gramado, como por exemplo, jogadores, bola e juiz.

Foram considerados outros detectores de arestas existentes na literatura, como os de Canny [Canny86], Sobel [Gonzales+93] e Perona-Malik [Perona+90]. Os resultados obtidos foram semelhantes, no entanto, ao obtidos com o filtro *LoG*. Optou-se, então, por este último por sua simplicidade e por seu tempo de processamento.

Em relação ao processo de segmentação, pudemos notar que na maioria das vezes não é possível realizá-lo sem antes filtrar a imagem. Mesmo assim, quando é feita pelo método de segmentação por limiar, esta pode ser prejudicada se a filtragem não foi muito bem realizada. Outro problema relativo à segmentação por limiar está em encontrar um valor de corte ideal, que muitas vezes é impossível de ser calculado de forma automática. Alguns métodos existentes na literatura, como o método de Otsu, resolvem para muitos casos mas não para todos.

Finalmente, podemos concluir que o melhor método encontrado para realçar e extrair os pontos de interesse foi seguir os seguintes passos:

1. Converter a imagem em negativo através da transformação negativa quando os pontos que desejamos realçar são mais claros que seus vizinhos;
2. Filtrar a imagem com o filtro *LoG*;
3. Aplicar o processo de segmentação através da utilização de um limiar de corte, mas, por enquanto, sem um método automático para determinar um valor ótimo de corte.

Capítulo 3

3 Extração de Segmentos de Retas

Neste capítulo, são abordados alguns algoritmos para a extração de segmentos de retas de uma imagem, na qual supomos que já foram identificados pontos candidatos a estarem sobre algum segmento de reta, através do estágio de pré-processamento em baixo nível, abordado no capítulo anterior. O próximo objetivo consiste em localizar e extrair esses possíveis segmentos, conforme as formas descritas em uma base de conhecimento, que caracterizam o conjunto das linhas de um modelo, como, por exemplo, de um campo de futebol. Como resultado do estágio de pré-processamento, o que existem são apenas informações sobre pontos que são candidatos a estarem sobre algum segmento de reta. O resultado desejado, no entanto, deve ser dado por uma estrutura geométrica – mais explicitamente, parâmetros que definam esses segmentos ou pontos que os delimitam.

Existem diversos algoritmos que, dado um conjunto de pontos, encontram retas ou segmentos de retas que mais se adaptam a ele. Uma possível solução é primeiro encontrar todas as linhas determinadas por qualquer par de pontos e então encontrar todos os subconjuntos de pontos que estejam próximos de uma linha particular. O problema para este procedimento é que envolve encontrar $\frac{n(n-1)}{2} \approx n^2$ linhas e então realizar $\frac{(n)(n(n-1))}{2} \approx n^3$ comparações para cada ponto para todas as linhas, o que o torna computacionalmente muito ineficiente.

Na tentativa de encontrar algoritmos melhores, diversos pesquisadores procuram soluções com menos esforço computacional. O mais conhecido entre os algoritmos descritos

na literatura é a transformada de Hough [Gonzales+93], que localiza retas, ou outras curvas paramétricas, a partir de um espaço de parâmetros gerado. Este algoritmo é descrito brevemente na próxima seção deste capítulo, enumerando as características contra e a favor e exemplificando com algumas imagens de testes.

Em seguida, é apresentado um novo algoritmo proposto nesta tese, que trabalha a partir de divisões da imagem, tendo seus passos descritos em detalhes e sendo comparado com a transformada de Hough. Vale lembrar que, no caso específico desta tese, estamos interessados em obter segmentos de reta longos. Essa característica é essencial para o algoritmo proposto, pois assim podemos descartar de forma fácil e eficiente pontos que foram detectados pela filtragem mas que estão sobre alguns objetos de interferência. No caso da visualização de uma partida de futebol, os objetos de interferência podem ser jogadores, juiz, bola, bandeirinhas, cartazes ao redor do campo, público da arquibancada ou então ruídos gerados pela interferência do sinal de transmissão da imagem de televisão.

3.1 Transformada de Hough

Em 1962, Hough [Hough62] propôs um método para reconhecer padrões chamado transformada de Hough, e Duda e Hart, em [Duda+72], aplicaram esta transformada para detectar um conjunto de linhas a partir de um conjunto de pontos utilizando um espaço diferente da imagem, chamado de espaço de parâmetros.

A formulação da equação de reta utilizada nesta tese na transformada de Hough é descrita por

$$u.\cos(\theta) + v.\sin(\theta) = \rho \tag{3.1}$$

onde θ e ρ são constantes e indicam a inclinação da reta e a distância da reta à origem do sistema, respectivamente, conforme podemos ver na Figura 3.1.

Com a equação (3.1), temos que o espaço de parâmetros gerado pela transformada de Hough é dado pelo plano de coordenadas θ e ρ. O motivo de escrever a equação de reta dessa forma, em vez da forma mais comum, descrita por

$$v = au + b \tag{3.2}$$

onde a e b são constantes, é que assim podemos limitar o espaço de parâmetros gerado pela transformada de Hough, onde θ varia de -90° a +90° e ρ varia de 0 a $\sqrt{2}d$, onde d é o

comprimento da diagonal do retângulo da imagem. Com esta escolha, cada ponto da imagem descreve uma curva senoidal no espaço de parâmetros. Na forma tradicional, cada ponto da imagem é representado por uma reta no espaço de parâmetros, descrito no plano *ab*.

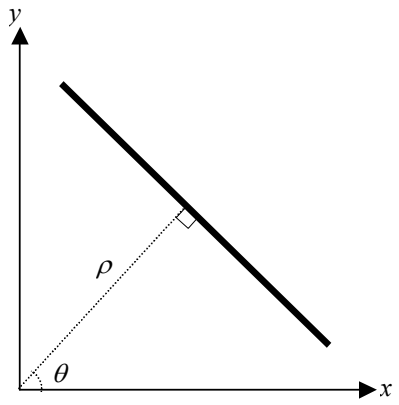


Figura 3.1 – Parâmetros que descrevem uma reta.

Cada interseção de duas curvas no espaço de parâmetros fornece uma reta que passa pelos dois pontos na imagem que representam tais curvas.

Quanto mais existirem pontos selecionados na imagem como candidatos a estarem sobre algum segmento de reta, mais curvas teremos no espaço de parâmetros, e com isto mais interseções ocorrerão, resultando em mais linhas encontradas na imagem. Para não obter todas as linhas que passam por quaisquer dois pontos na imagem, determinamos apenas os *n* pontos de maiores interseções no espaço de parâmetros, tendo assim apenas *n* linhas obtidas na imagem. O problema é que nem sempre é possível determinar qual o melhor valor para *n*.

Quando a transformada de Hough é implementada, geralmente modela-se o espaço de parâmetros na forma de uma matriz de acumulação, iniciada com todos os valores iguais a zero, devido à necessidade de discretizar a implementação. Cada elemento dessa matriz indica o número de curvas que passam pelo ponto correspondente a este elemento, isto é, indicam o número de interseções das curvas em um dado ponto.

Para exemplificar, o método da transformada de Hough foi aplicado a uma imagem artificial de um esquema de um campo de futebol, onde só são visualizados segmentos de retas (linhas do campo de futebol), conforme ilustra a Figura 3.2. A imagem que representa o espaço dos parâmetros relativo a esta imagem é vista na Figura 3.3. O resultado final do método aplicado à Figura 3.2 são as linhas representadas com a cor vermelha na Figura 3.4.

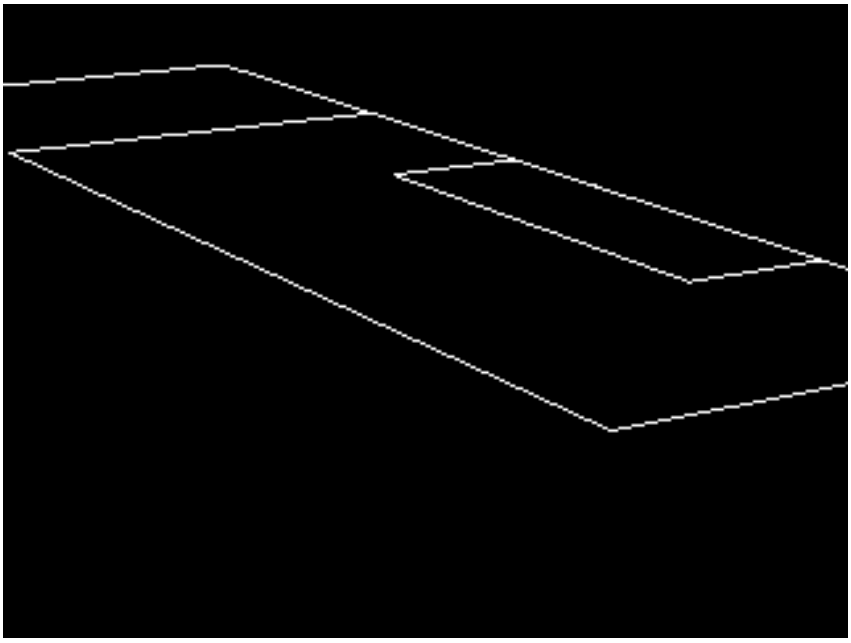


Figura 3.2 – Imagem artificial de um conjunto de segmento de retas.

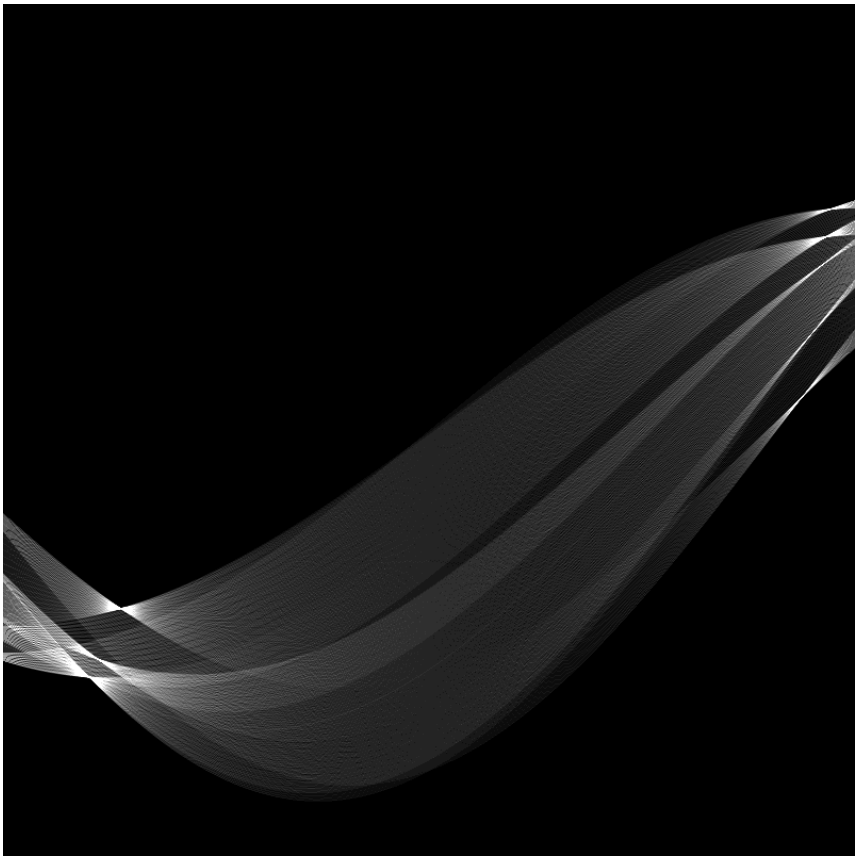


Figura 3.3 – Espaço de parâmetros relativo à Figura 3.2.

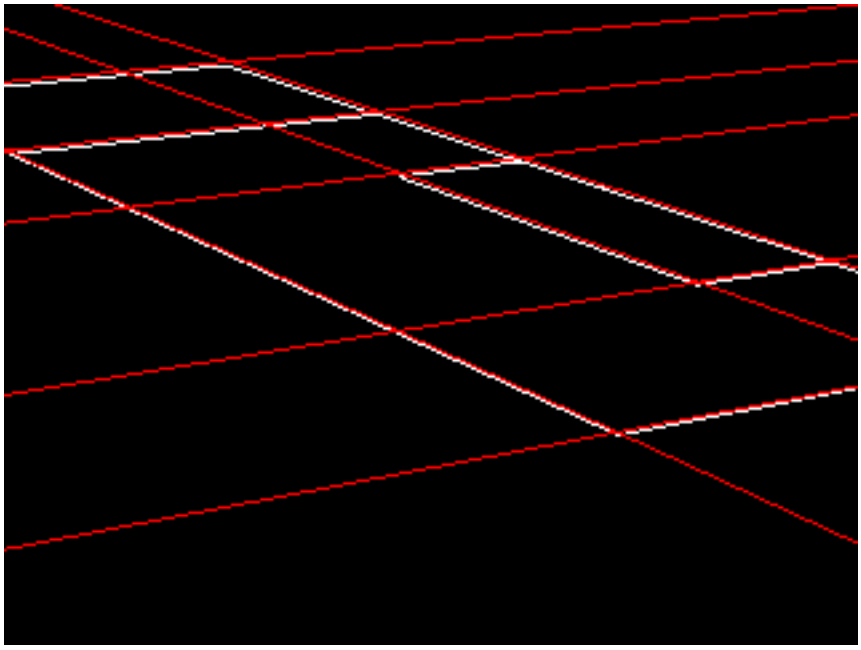


Figura 3.4 – Resultado do método da transformada de Hough aplicado à Figura 3.2.

Podemos observar que algumas linhas encontradas têm pequenas diferenças de localização para as linhas que deveriam ser realmente encontradas. Estes erros se devem a pequenas variações de ρ e θ, que, por sua vez, ocorrem em virtude da quantização realizada no espaço de parâmetros. Porém, desconsiderando estes pequenos erros, podemos dizer que as retas suporte dos segmentos de retas existentes na imagem foram localizadas.

Uma grande desvantagem desse método é a lentidão (complexidade computacional), pois para cada ponto da imagem devemos atualizar toda a matriz que representa o espaço de parâmetros.

No caso de uma imagem de uma cena real, existe ainda um problema bem maior, que ocorre quando temos vários pontos que não formam retas, isto é, formam apenas uma região não bem definida. Neste caso, o método pode retornar linhas inexistentes na imagem, como mostra o exemplo abaixo, onde existem diversos pontos brancos localizados na região da arquibancada. O método da transformada de Hough foi aplicado à Figura 3.5, obtendo o espaço de parâmetros ilustrado na Figura 3.6. O resultado está na Figura 3.7, na qual as linhas obtidas estão desenhadas em vermelho. Observe a quantidade de linhas obtidas que não correspondem a linhas do campo, enquanto que diversas linhas do campo não foram detectadas.

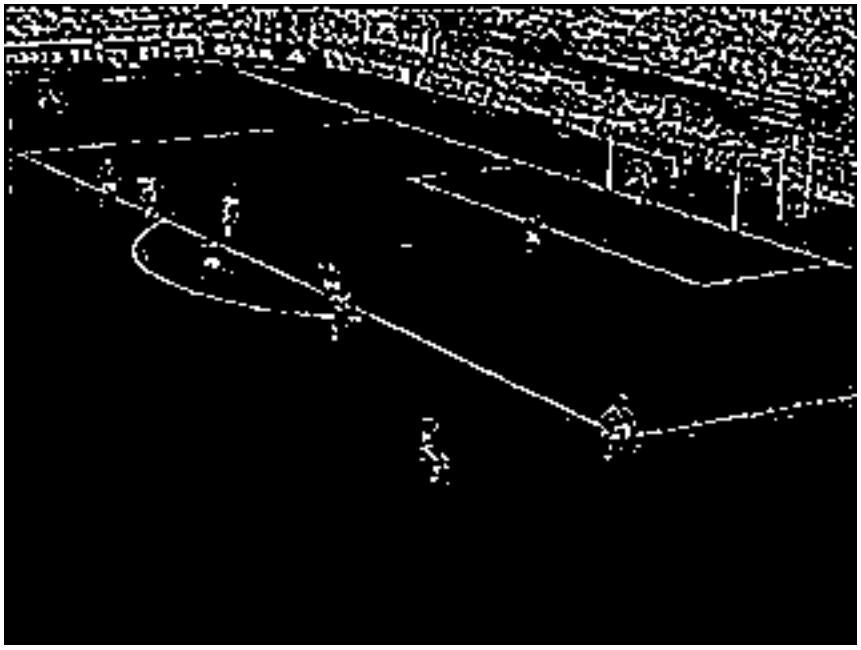


Figura 3.5 – Imagem de entrada para o método de Hough um pouco mais realista.

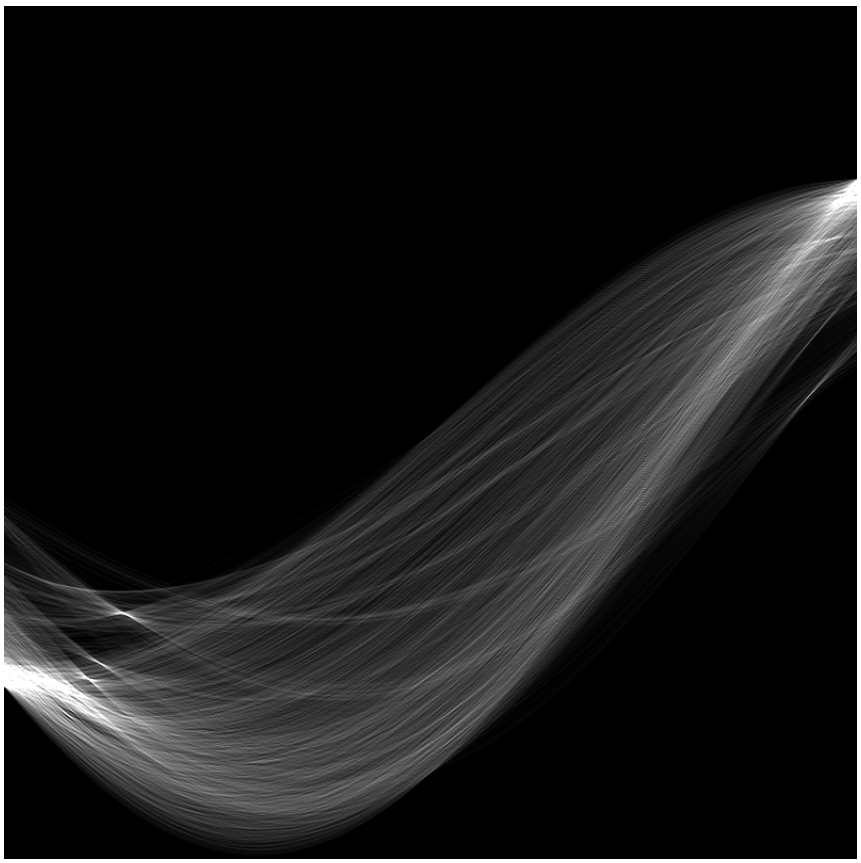


Figura 3.6 – Espaço de parâmetros da transformada de Hough.

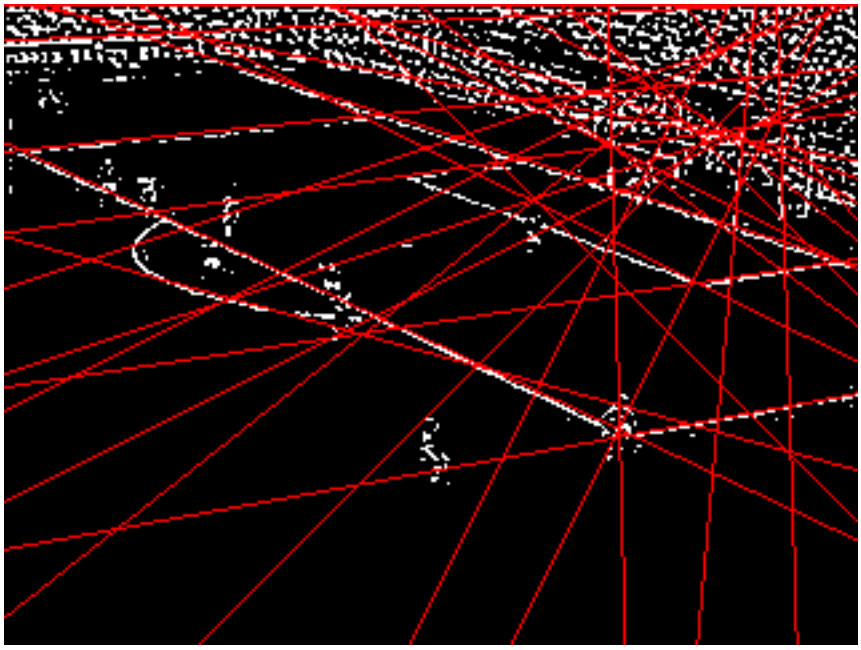


Figura 3.7 – Resultado do método de Hough.

O problema de encontrar linhas onde não existem, o que pode ser devido à quantização feita no espaço de parâmetros, pode ocorrer também porque este método é tratado de forma pontual na imagem, isto é, a "construção" do espaço de parâmetros é realizada sem levar em consideração vizinhanças dos pontos.

Mesmo se fossem removidos alguns pontos brancos na Figura 3.5, como os que representam o público na arquibancada, retirados na Figura 3.8, o resultado da transformada de Hough ainda não seria satisfatório, conforme visto na Figura 3.9.

O resultado apresentado na Figura 3.9 deve-se ao fato de existirem jogadores na imagem e das linhas do campo não serem verdadeiramente retas devido a diversos fatores, como a distorção radial da lente da câmera que gerou a imagem. Alguns dos problemas que na prática também dificultam a localização de linhas através desse método são a espessura das linhas, acarretando múltiplas linhas, e o fato de que os pontos da imagem são dados por coordenadas inteiras.

Os problemas descritos acima, somados à pouca eficiência em tempo de processamento levaram-nos a buscar uma alternativa à transformada de Hough, descrita nas seções a seguir. Entretanto, uma grande vantagem deste método é a facilidade de estendê-lo, além de retas, para outros tipos de primitivas geométricas com equação matemática definida,

como, por exemplo, círculos, obtendo outros tipos de espaços de parâmetros mas mantendo sempre a mesma base do algoritmo. Por outro lado, a complexidade aumenta junto com o tempo de processamento. Como é abordado no próximo capítulo, este método pode ser modificado para resolver problemas de reconhecimento e interpretação.

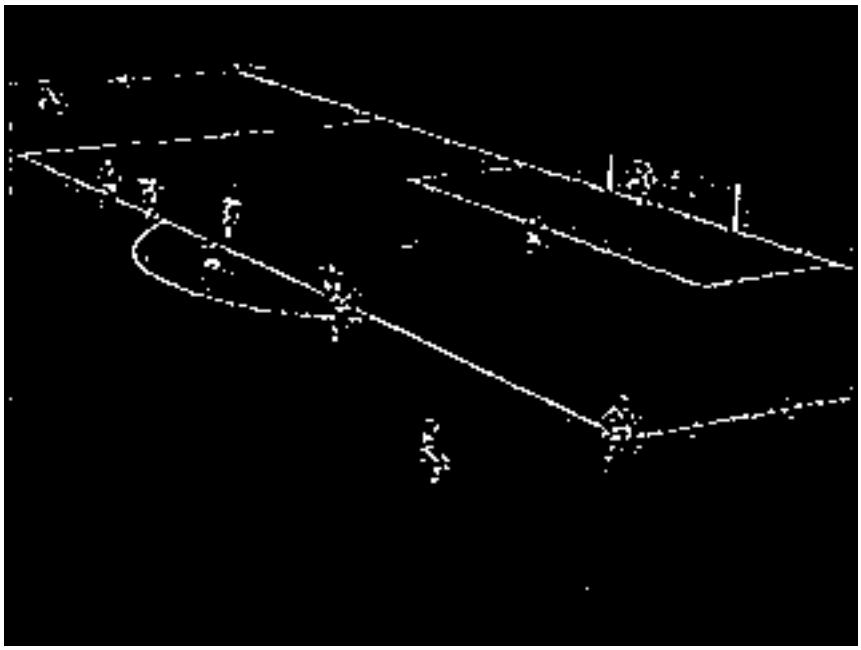


Figura 3.8 – Imagem sem arquibancada.

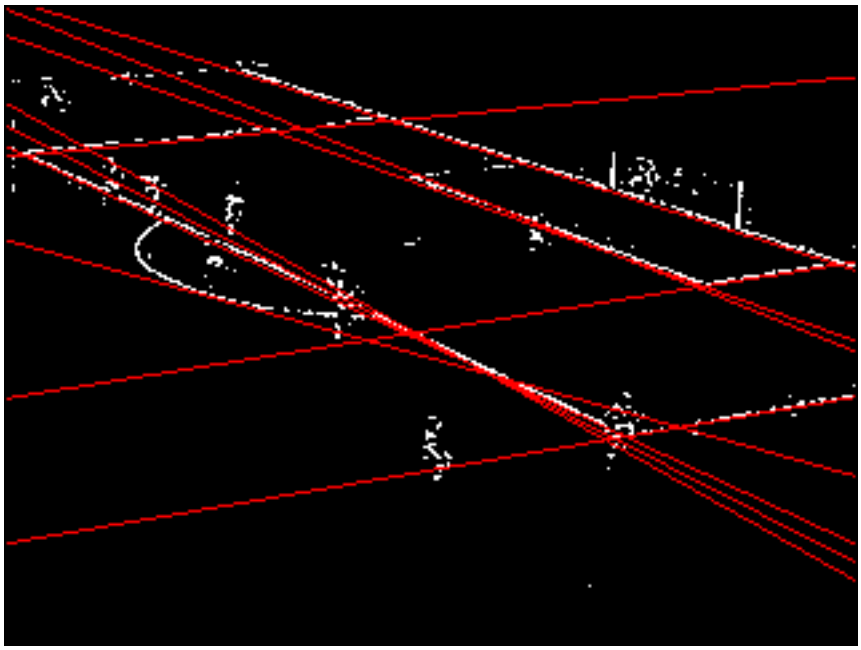


Figura 3.9 – Linhas encontradas através da transformada de Hough relativa à Figura 3.8.

3.2 Algoritmo proposto

No primeiro passo do algoritmo proposto, são eliminados os pontos que não estão sobre nenhum segmento de reta ou que estão sobre um segmento de reta muito curto. Em seguida são determinados os segmentos de retas formados pelos pontos que não foram descartados. No final, um passo de reajuste é dado aos segmentos. Este passo é de fundamental importância para o algoritmo de acompanhamento de cenas proposto nesta tese, principalmente para obter eficiência no desempenho.

3.2.1 Eliminação de Pontos

A primeira etapa do algoritmo proposto visa eliminar os pontos que não estão sobre nenhum segmento de reta contido na imagem. A imagem de entrada para esta etapa é a resultante da filtragem *LoG* seguida da aplicação de uma segmentação por limiar. Para a eliminação desses pontos, a imagem é dividida em células retangulares de iguais dimensões, segundo uma grade regular, como mostrado na Figura 3.10.

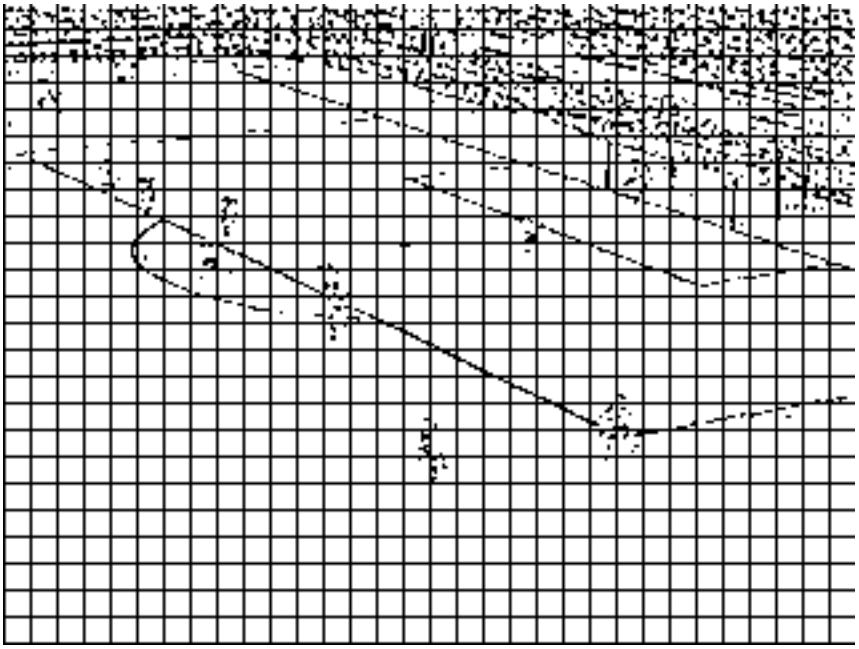


Figura 3.10 – Imagem dividida em células segundo uma grade regular.

Esta figura, assim como outras que irão aparecer, tem os pontos resultantes das filtragens com cor invertida, isto é, as imagens aparecem na sua forma negativa, apenas para realçar melhor visualmente as células e os pontos que estão contidos.

Note-se que algumas células da Figura 3.10 não contêm nenhum ponto. Nelas não há nada para ser feito. Entretanto, em outras células, existem diversos pontos, mas muito espalhados, não indicando a existência de nenhuma direção predominante de sua disposição no interior da célula. Os pontos contidos nessas células também não são do nosso interesse, pois não formam um segmento de reta. Finalmente, existe ainda outro conjunto de células, nas quais os pontos estão dispostos sobre uma reta. Nelas, os pontos indicam a existência de uma direção predominante de disposição. Podemos notar visualmente que nestas células estão os pontos que encontram-se sobre alguma linha do campo, mas também há casos em que estão sobre um jogador ou algum outro objeto.

A partir dessa classificação, dividimos as células em dois grupos. No primeiro estão as células cujo conjunto de pontos não indica uma direção predominante, devendo, portanto, ser descartados. No segundo grupo estão as demais células, isto é, as células nas quais o conjunto de pontos indica uma direção predominante, formando um segmento de reta.

Esta classificação das células em dois grupos é dada a partir dos autovalores λ_1 e λ_2, referentes à matriz de covariância ([Andrews91] e [Gonzales+93]) calculada para cada célula. Esta matriz é dada por:

$$\begin{bmatrix} a & b \\ b & c \end{bmatrix} \tag{3.3}$$

onde

$$a = \frac{\sum_{i=1}^{n} (u_i - \bar{u})^2}{n} \tag{3.4}$$

$$b = \frac{\sum_{i=1}^{n} (u_i - \bar{u})(v_i - \bar{v})}{n} \tag{3.5}$$

$$c = \frac{\sum_{i=1}^{n} (v_i - \bar{v})^2}{n} \tag{3.6}$$

$$(\bar{u}, \bar{v}) = \frac{\sum_{i=1}^{n} (u_i, v_i)}{n} \tag{3.7}$$

sendo n o número de pontos pretos na célula e u_i e v_i as coordenadas cartesianas de cada ponto preto ($\overline{u}$ e $\overline{v}$, portanto, são as coordenadas cartesianas do centróide dos pontos pretos).

Através dos autovalores λ_1 e λ_2 da matriz de covariância (3.3), dados por:

$$\lambda_1 = \frac{a + c + \sqrt{(a-c)^2 + 4b^2}}{2} \tag{3.8}$$

e

$$\lambda_2 = \frac{a + c - \sqrt{(a-c)^2 + 4b^2}}{2} \tag{3.9}$$

podemos quantificar o quanto os pontos pretos em uma célula estão posicionados sobre uma reta. Podemos notar que $\lambda_1 \geq \lambda_2$ e, pelas propriedades da matriz de covariância, esses dois autovalores são sempre números reais não-negativos. Quanto maior for a razão entre o maior e o menor autovalor, mais o conjunto dos pontos pretos se estende ao longo de uma direção, que é determinada pelo autovetor relativo ao maior autovalor (λ_1).

A Figura 3.11 ilustra a existência da direção predominante de um conjunto de pontos, onde v_{λ_1} e v_{λ_2} são os autovetores correspondentes aos autovalores λ_1 e λ_2, respectivamente.

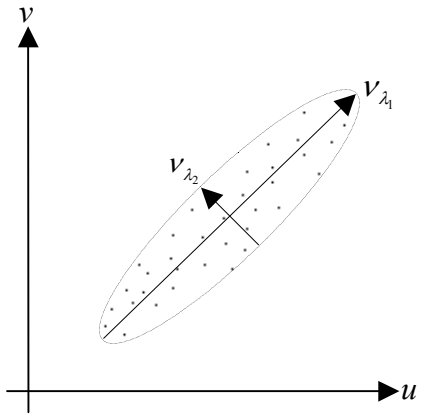


Figura 3.11 – Análise de componentes principais.

O método para encontrar estas direções predominantes com base na matriz de covariância é conhecido como análise de componentes principais.

Podemos, portanto, classificar estas células da seguinte forma:

- Se a razão entre o maior e o menor autovalor for menor que um determinado valor e sendo os dois autovalores diferentes de zero, os pontos destas células também

são todos descartados. Este é o caso das células que estão sobre a visualização da região da arquibancada da Figura 3.10.

- Caso contrário, se a razão entre o maior e o menor autovalor for maior que um determinado valor, ou se apenas um dos autovalores for nulo, então os pontos destas células não são descartados e a direção predominante da disposição dos pontos é dada pelo autovetor (v_{λ_1}) correspondente ao maior autovalor (λ_1). Este é o caso das células que contêm as linhas do campo, mas podem conter também jogadores em alguns casos.

O resultado desta classificação é visto na Figura 3.12.

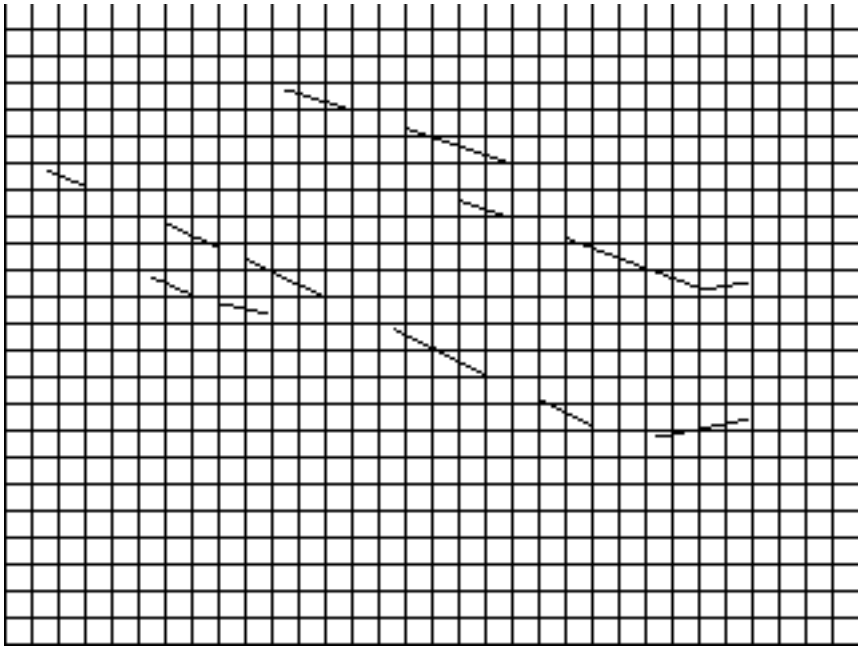


Figura 3.12 – Resultado da classificação das células.

Podemos observar que na Figura 3.12 existem diversas descontinuidades. Isto se dá por diversos motivos, como por exemplo:

- existem jogadores sobre algumas linhas do campo, fazendo com que nestas células todos os pontos tenham sido descartados por não existir uma direção predominante;
- as linhas do campo em certas partes não são bem visíveis, então a filtragem *LoG* junto com a segmentação por limiar descartaram alguns pontos que poderiam estar sobre uma linha do campo;

- ou ainda há ruídos que não foram, e deveriam ter sido, eliminados durante a filtragem *LoG*, provenientes da transmissão e captura da imagem por uma rede de televisão.

Da maneira como a técnica foi exposta até agora, todos os pontos têm mesmo peso, isto é, interferem da mesma forma para determinar se uma célula possui ou não uma direção predominante e qual direção é esta, caso exista. Porém, muitas vezes, o valor de um ponto contém informações importantes e que poderiam ser usadas com um certo peso. Exemplos dessas informações são:

- resultado da filtragem *LoG*;
- luminância;
- ou uma combinação desses dois.

Suponha que um ponto (u_i, v_i) tenha um peso κ_i. Levando em consideração este peso, os valores da matriz de covariância, dada em (3.3), são calculados através das seguintes equações:

$$a = \frac{\sum_{i=1}^{n} \left(\kappa_i \left(u_i - \overline{u} \right)^2 \right)}{\sum_{i=1}^{n} \kappa_i} \tag{3.10}$$

$$b = \frac{\sum_{i=1}^{n} \left(\kappa_i \left(u_i - \overline{u} \right) \left(v_i - \overline{v} \right) \right)}{\sum_{i=1}^{n} \kappa_i} \tag{3.11}$$

$$c = \frac{\sum_{i=1}^{n} \left(\kappa_i \left(v_i - \overline{v} \right)^2 \right)}{\sum_{i=1}^{n} \kappa_i} \tag{3.12}$$

$$\left(\overline{u}, \overline{v} \right) = \frac{\sum_{i=1}^{n} \left(\kappa_i \left(u_i, v_i \right) \right)}{\sum_{i=1}^{n} \kappa_i} \tag{3.13}$$

As vantagens de utilizar pesos estão relacionadas a:

- poder relaxar a segmentação feita pelo algoritmo de segmentação por limiar, atribuindo valores menores ao limiar de corte;
- com essa relaxação, mais pontos influenciarão o cálculo da matriz de covariância, utilizando mais informações da imagem original.

Entre as desvantagens, estão:

- maior tempo de processamento;
- inclusão de ruídos, ou seja, se a relaxação for grande, muitos pontos que foram descartados na segmentação devido a ruídos presentes na imagem farão parte do cálculo da matriz de covariância, podendo surgir linhas inexistentes na imagem.

Devido à primeira desvantagem listada acima, como deseja-se obter um processamento em tempo real, a opção de usar pesos no algoritmo proposto nesta tese foi descartada.

3.2.2 Determinação dos Segmentos de Retas

Por enquanto, temos apenas resultados em pontos. Nesta seção é visto como encontrar os segmentos de retas, obtendo sua estrutura geométrica a partir dos resultados da etapa anterior. Em alguns modelos, como na região da grande área em um campo de futebol, não existem segmentos disjuntos colineares, portanto os segmentos de retas encontrados para estes modelos devem ser de modo que dois segmentos de retas não possam estar sobre a mesma reta suporte.

No primeiro passo deste algoritmo, são atribuídos valores a cada célula, de forma que se duas células tiverem o mesmo valor elas conterão direções predominantes próximas quando seus pontos são coletados em um mesmo conjunto. Em seguida são extraídos os segmentos de retas, através do método de mínimos quadrados, a partir dos pontos das células de mesmos valores. Para finalizar, são conectados os segmentos que estão sobre a mesma reta suporte, pelo motivo descrito no parágrafo anterior, quando necessário.

3.2.2.1 Atribuição de Valores às Células

Para atribuir valores a cada célula, as células são percorridas de modo que as linhas de células são processadas de baixo para cima e, em cada linha, as células são processadas da esquerda para a direita.

O valor atribuído a cada célula é um número inteiro determinado comparando os autovetores v_{λ_1} da célula corrente e das células vizinhas já visitadas – uma a esquerda e três abaixo, como mostrado na Figura 3.13 (a célula hachurada representa a célula corrente e está apontando para as células vizinhas já visitadas, uma vez que as células são percorridas de baixo para cima, da esquerda para a direita).

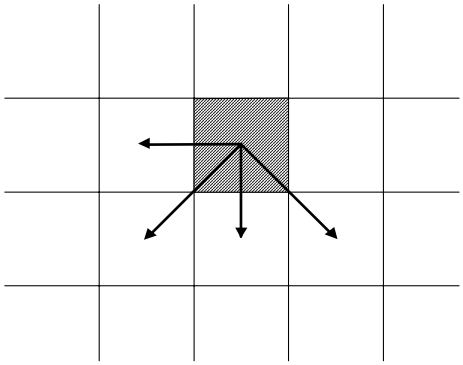


Figura 3.13 – Células vizinhas consultadas.

Os valores de cada célula são dados da seguinte forma:

- se a célula corrente não possui direção predominante, então seu valor é zero. Não há necessidade de consultar nenhuma célula;
- caso contrário, é verificado se existe alguma célula vizinha que já tenha um valor atribuído e que, se juntarmos seus pontos com os da célula corrente, teremos uma direção predominante próxima à da célula corrente. Caso não exista esta célula com direção próxima, um novo valor é dado à célula corrente; senão, seu valor é o mesmo da célula que ofereça a direção mais próxima à direção da célula corrente quando unidas.

No segundo item acima, é importante observar que a direção a ser comparada com a direção da célula corrente é fornecida pela união dos pontos das duas células – a célula corrente e a que está sendo consultada – pois, caso fosse feita a comparação diretamente sem a união, poderíamos obter resultados errados, uma vez que os autovetores indicam apenas a

direção de uma possível reta suporte. Um exemplo que acarretaria este erro é se tivermos duas células vizinhas com os pontos dispostos em duas retas suporte distintas mas paralelas entre si, uma em cada célula. As duas células têm a mesma direção predominante, embora isso não ocorra quando juntas.

Se visualizarmos agora as células ilustradas na Figura 3.12, mas com o resultado da atribuição de valores indicado por cores, teremos a imagem ilustrada na Figura 3.14. As células em branco indicam que seus valores são iguais a zero. No nosso exemplo, existem 14 valores diferentes atribuídos às células.

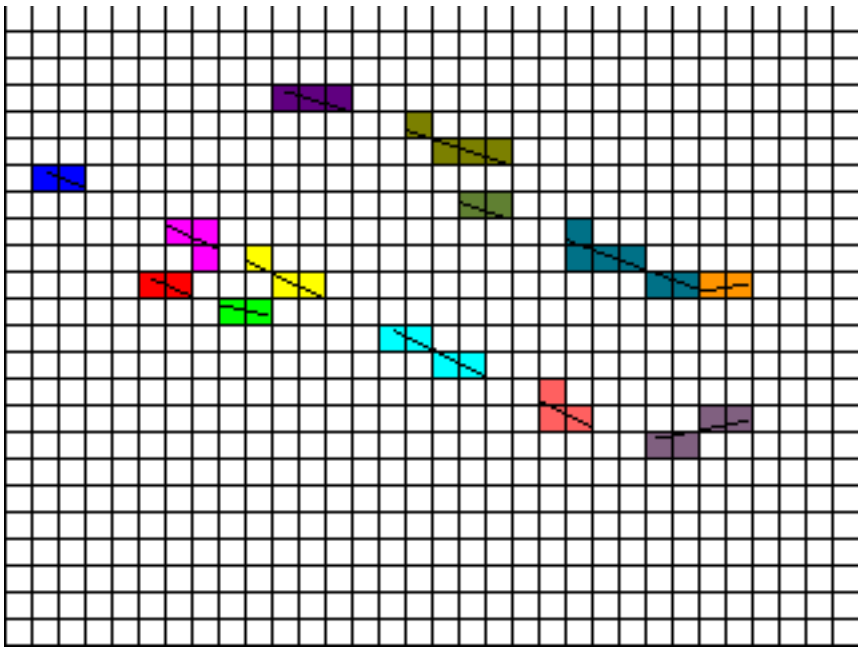


Figura 3.14 – Células com os valores atribuídos indicados com cores.

3.2.2.2 Extração dos Segmentos de Retas

A extração dos segmentos de retas é feita para cada conjunto de células de iguais valores. No exemplo da Figura 3.14 são extraídos 13 segmentos, pois há 14 valores diferentes atribuídos às células da imagem, sendo que o valor 0 indica ausência de segmento de reta.

Para extrair os segmentos, é usado o método de mínimos quadrados para determinar cada reta seguido por um procedimento de recorte para limitá-la a um segmento. Poderiam ser utilizados os autovetores já calculados para obter as direções dos segmentos a serem extraídos e o centróide como sendo um ponto pertencente à reta. Porém, pelo fato de não termos utilizado pesos nos pontos na obtenção dos autovetores e por estes pesos serem

importantes na determinação das retas, não são utilizados os autovetores, mas um algoritmo de mínimos quadrados onde cada ponto (u_i, v_i) tem um peso κ_i.

Através do método de mínimos quadrados e supondo a equação da reta descrita por $v = au+b$, temos que

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} u_i^2 & \sum_{i=1}^{n} u_i \\ \sum_{i=1}^{n} u_i & n \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^{n} u_i v_i \\ \sum_{i=1}^{n} v_i \end{bmatrix} \tag{3.14}$$

Caso seja atribuído um peso κ_i a cada ponto, a e b são dados por

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} \kappa_i u_i^2 & \sum_{i=1}^{n} \kappa_i u_i \\ \sum_{i=1}^{n} \kappa_i u_i & \sum_{i=1}^{n} \kappa_i \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^{n} \kappa_i u_i v_i \\ \sum_{i=1}^{n} \kappa_i v_i \end{bmatrix} \tag{3.15}$$

Como, nas regras do futebol, as linhas do campo são pintadas de branco, isto resulta que os pontos da imagem que as representam tenham valores de luminância maiores que seus pontos vizinhos (supostamente pontos do gramado), resultando também em maiores valores obtidos na filtragem *LoG* destes pontos. Isto nos leva a deduzir que os valores obtidos na filtragem *LoG* são muito importantes para a determinação dos segmentos de retas: eles podem ser fornecidos como pesos para o algoritmo de mínimos quadrados. Isto é, terão maior influência na determinação das retas os pontos com maiores valores de laplaciano obtidos pela filtragem *LoG*. Ou seja, adotamos como os pesos κ_i os valores dos pontos da imagem resultante da filtragem *LoG*.

Para limitar as retas, simplesmente calcula-se a caixa envoltória dos pontos que as originaram, ou seja, determina-se a caixa envoltória a partir dos limites inferiores e superiores nas direções $\overrightarrow{ou}$ e $\overrightarrow{ov}$. O segmento resultante é dado pela reta recortada por estes valores, representado pelos seus extremos. A Figura 3.15 ilustra este recorte, onde o retângulo de bordas tracejadas representa a caixa envoltória dos pontos, a reta em diagonal fina é o resultado do método de mínimos quadrados e o segmento espesso a sua limitação dada pela caixa envoltória.

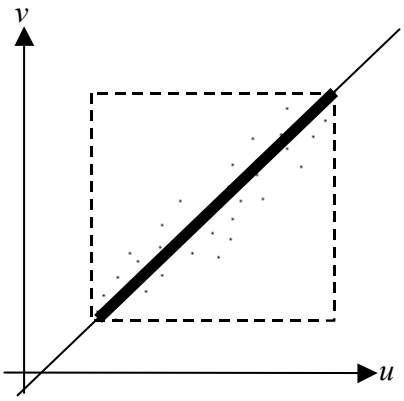


Figura 3.15 – Caixa envoltória dos pontos limitando uma reta e gerando um segmento relativo à Figura 3.11.

3.2.2.3 União dos Segmentos de Retas

Como já foi mencionado, em uma região do campo de futebol localizada do meio do campo até um dos gols, não existem dois segmentos sobre uma mesma reta suporte, isto é, não há descontinuidade dos segmentos. Se um outro modelo utilizado não apresentar esta mesma característica do campo de futebol, então a união dos segmentos não deverá ser realizada. Uma outra alternativa é utilizar um limiar de distância, abaixo da qual segmentos são unidos.

As linhas do campo de futebol apresentam-se apenas segundo duas direções mutuamente ortogonais. Porém, a visualização dessas linhas através de uma câmera pode distorcer um pouco essas características pelos seguintes motivos:

1. Distorções radiais causadas pela lente da câmera: neste caso uma linha do campo pode apresentar uma curvatura suave. Isto é, os segmentos que, quando unidos, formam um novo segmento, têm pequenas variações angulares e disposições quando sobrepostos ao novo segmento. Exemplos disto podem ser vistos na Figura 3.16, com os segmentos 2, 3, 6, 9 e 11, ou os segmentos 7 e 10, ou ainda os segmentos 12 e 13.
2. Transformação projetiva: aqui as linhas do campo que no caso real são paralelas podem não o ser na visualização, apresentando um pequeno ângulo entre elas. O mesmo acontece com as linhas ortogonais: na visualização, podem não ser ortogonais, mas ainda apresentam um ângulo grande. Um exemplo pode ser visto na Figura 3.16, com os segmentos 7 e 8.

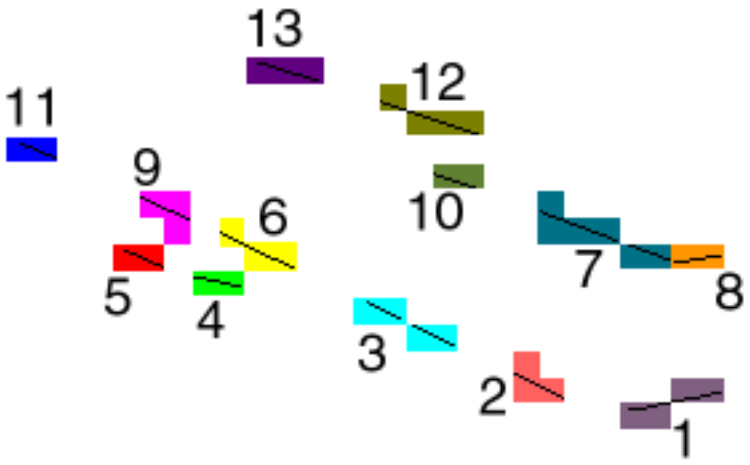


Figura 3.16 – Segmentos numerados a serem unidos.

Algumas pequenas deformações no modelo também podem ser apresentadas. Um exemplo disso está na Figura 7.15, na qual o modelo está impresso em uma folha de papel acomodada sobre um cadeira, apresentando algumas dobras que deformam o modelo impresso.

Estas caracterizações vão auxiliar na união dos segmentos: segmentos que apresentam ângulos pequenos e que estão localizados aproximadamente sobre uma mesma reta suporte devem ser unidos, conforme ilustra a Figura 3.17. A verificação se dois segmentos, $\overline{ab}$ e $\overline{cd}$, apresentam um ângulo pequeno é feita através da norma do produto interno dos vetores $\overrightarrow{ab}$ e $\overrightarrow{cd}$ dividido pelo produto de suas normas, ou seja,

$$\alpha = \frac{\left|\overrightarrow{ab}.\overrightarrow{cd}\right|}{\left\|\overrightarrow{ab}\right\|.\left\|\overrightarrow{cd}\right\|} \tag{3.19}$$

Se o valor de α for menor que um dado valor, então é encontrado um novo segmento de reta $\overline{ef}$ a partir da união dos pontos que originaram os segmentos $\overline{ab}$ e $\overline{cd}$, utilizando o método de mínimos quadrados conforme descrito na Seção 3.2.2.2. Se os pontos *a*, *b*, *c* e *d* estiverem próximos ao novo segmento, então $\overline{ef}$ substituirá $\overline{ab}$ e $\overline{cd}$.

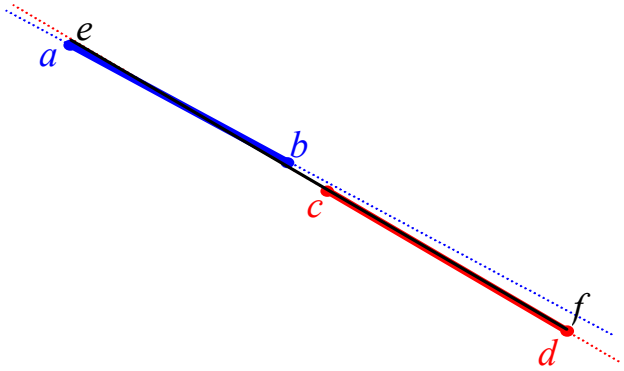


Figura 3.17 – União de segmentos de retas.

O resultado do algoritmo de união aplicado ao exemplo da Figura 3.14, é o conjunto de segmento de retas enumerados na Figura 3.18, com as descrições de cada segmento, informando as coordenadas de seus pontos extremos na Tabela 3.1. Na Figura 3.19 estes segmentos estão sobrepostos à imagem original (Figura 2.1).

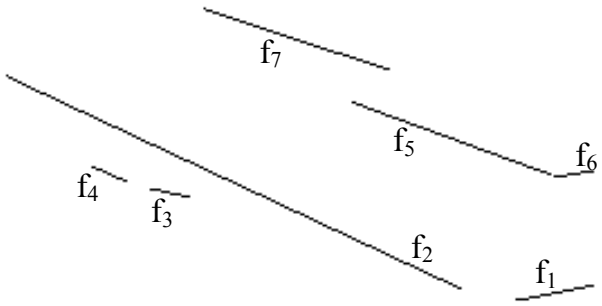


Figura 3.18– Segmentos unidos extraídos da Figura 2.1, enumerados conforme resultado do método.

Tabela 3.1 – Coordenadas dos segmentos extraídos, ilustrados na Figura 3.18.

f_1 - (243.699,79)-(278,85.6609)	f_2 - (16.6541,179)-(219,84.1056)
f_3 - (80,128.996)-(97.4564,125)	f_4 - (54,138.838)-(69,132.109)
f_5 - (170,167.472)-(258.807,135)	f_6 - (260.649,134)-(278.087,137)
f_7 - (104.579,209)-(186.959,182)	

Figura 3.19 – Sobreposição dos segmentos extraídos à imagem.

Observe que nem todas as linhas foram extraídas e algumas estão parciais. Isto deve-se à eliminação dos pontos, resultantes da filtragem *LoG* seguido de uma segmentação, contidos em células que não apresentam pontos dispostos sobre um segmento de reta (Seção 3.2.1).

3.2.3 Reajuste das Linhas

Uma vez detectado que algumas linhas visualizadas são referentes às linhas do campo de futebol ou fazem referência a quaisquer outros objetos presentes na imagem como jogadores, uma pergunta imediata e lógica pode ser feita: o quanto estas linhas foram bem extraídas da imagem?

Como o algoritmo descarta diversos pontos ao mesmo tempo quando estes se localizam em células, os quais *a priori* não teriam influência sobre a detecção de uma reta segundo a maneira proposta, pode ser que sejam descartados pontos importantes quando não pensamos mais em células, mas na imagem como um todo.

Por isto, é importante que um reajuste das linhas detectadas seja realizado, mas em um contexto geral, ou seja, sem que a imagem seja mais dividida em células. Nos capítulos seguintes desta tese, veremos que este reajuste é muito importante, principalmente quando tivermos um outro conjunto das linhas além das detectadas – o conjunto das linhas

reconstruídas através de uma transformação, que dará origem ao conjunto das linhas do modelo reconstruídas. Este reajuste também é fundamental quando o algoritmo proposto for aplicado a uma seqüência de imagens, e não mais a uma imagem.

Quando trabalhamos com uma imagem completa, devemos ter cuidado para que pontos que não são desejados, como, por exemplo, os referentes ao público da arquibancada, não influenciem este passo de reajuste. O reajuste não deverá alterar muito as posições das linhas detectadas. Com isto, os pontos da imagem responsáveis por este reajuste devem estar próximos às linhas detectadas, e um ponto que está mais próximo de uma determinada linha só deverá influenciar esta linha no reajuste, nunca devendo influenciar duas linhas. Para isto, são usadas faixas de tolerância aplicadas às linhas detectadas e um coletor de pontos também responsável por dividi-los em grupos, sendo cada grupo associado a uma linha. Uma faixa de tolerância é definida por um segmento de reta e um valor *t*: um ponto está nesta faixa se estiver a uma distância da reta suporte do segmento menor ou igual a *t* e sua projeção na reta suporte estiver no interior do segmento. Exemplos de faixas de tolerância são mostrados na Figura 3.20, desenhadas em vermelho.

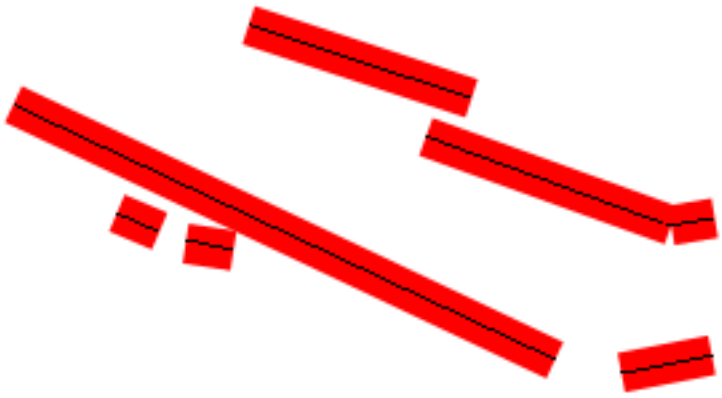


Figura 3.20 – Exemplo de faixa de tolerância.

Após terem sido coletados os pontos, determinam-se os novos segmentos de retas a partir de mínimos quadrados, encontrando retas e limitando-as conforme descrito na Seção 3.2.2.2.

3.3 Conclusão

Neste capítulo foram vistos dois algoritmos de extração de segmentos de retas em uma imagem já filtrada. O primeiro algoritmo, baseado na transformada de Hough, encontrava bem as linhas contidas nas imagens sintéticas que não apresentavam muitos pontos. É um algoritmo relativamente lento, o que inviabiliza seu uso no algoritmo global proposto nesta tese, que é de acompanhamento de cenas em tempo real. Ele apresentou também outros problemas relativos à precisão em localizar tais retas.

Com isto, foi necessário desenvolver um novo algoritmo, que atendesse, entre outros requisitos, à velocidade de extração dos segmentos. Este segundo algoritmo apresentou bons resultados mesmo quando a imagem, que foi assumida que já tinha sido filtrada, apresentava diversos pontos que não estavam sobre nenhuma reta. Trata-se de um algoritmo que processa inicialmente regiões isoladas eliminando diversos pontos indesejados e em seguida operando com regiões vizinhas, trabalhando com uma idéia de coerência. O algoritmo pode trabalhar tirando proveito de propriedades incorporadas às imagens, como o resultado da filtragem *LoG*.

Capítulo 4

4 Reconhecimento e Interpretação

A informação de saída que temos na etapa de extração de objetos, realizada no processamento em nível intermediário, é uma coleção de segmentos de retas. Alguns deles correspondem à visualização de segmentos das linhas do modelo, armazenado na base de conhecimentos, enquanto outros não têm correspondência com nenhuma linha do modelo, fazendo parte de outro contexto na imagem, como jogadores ou linhas de placas de propaganda existentes ao redor do campo.

Neste capítulo, são apresentados dois métodos de reconhecimento: o primeiro utiliza idéias semelhantes às da transformada de Hough e o segundo é um algoritmo baseado em uma árvore de interpretações. Além de reconhecer quais são os segmentos que pertencem à visualização das linhas do modelo, estes métodos sabem identificá-los, isto é, sabem indicar que uma tal linha é a visualização de uma determinada linha do modelo. Por exemplo, no modelo de linhas de um campo de futebol, eles estabelecem que um determinado segmento extraído da imagem representa a visualização de uma linha da lateral da grande área ou a linha de fundo.

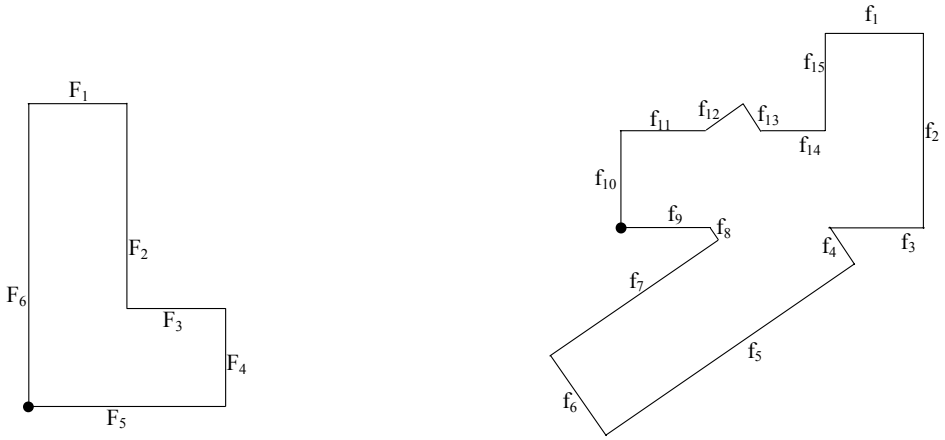
4.1 Transformada de Hough

Para facilitar o entendimento da técnica de reconhecimento utilizando a transformada de Hough [Grimson90], vamos supor que a visualização de um determinado modelo não sofreu nenhuma distorção ou escala, apenas uma rotação e translação. A inclusão de algumas dessas transformações apenas irá aumentar a complexidade do método. Neste método, pode

ocorrer de duas ou mais linhas da visualização representarem trechos diferentes de uma mesma linha do modelo, e também que nem todas as linhas do modelo estejam presentes na visualização e vice-versa.

A idéia principal deste método é semelhante ao método de encontrar retas em uma imagem utilizando a transformada de Hough, conforme apresentado no Capítulo 3. A transformada de Hough determinará um mapeamento de cada linha do modelo com as linhas da visualização, sendo este mapeamento caracterizado como um ponto no espaço de parâmetros. Neste caso, o espaço de parâmetros tem três dimensões: um valor escalar para o ângulo de rotação e uma translação dada por um vetor bidimensional. De forma análoga ao caso de encontrar retas, sempre incrementa-se um contador em cada ponto do espaço de parâmetros que certifica a transformação correta de uma linha do modelo a uma linha encontrada na visualização correspondente.

Para demonstrar o funcionamento e facilitar a compreensão do método, suponhamos que o modelo que temos seja o dado na Figura 4.1 (a), visualizado segundo a imagem da Figura 4.1 (b).



(a) Modelo (b) Visualização

Figura 4.1 – Exemplo de modelo e visualização.

Os dois círculos pretos nas figuras acima indicam as origens dos sistemas de coordenadas em cada figura.

Imaginemos que a linha F_1 do modelo tenha representação na visualização como a linha f_1. Isto significa que entre essas duas linhas há apenas uma translação e não há uma rotação. Graficamente, temos o resultado da Figura 4.2.

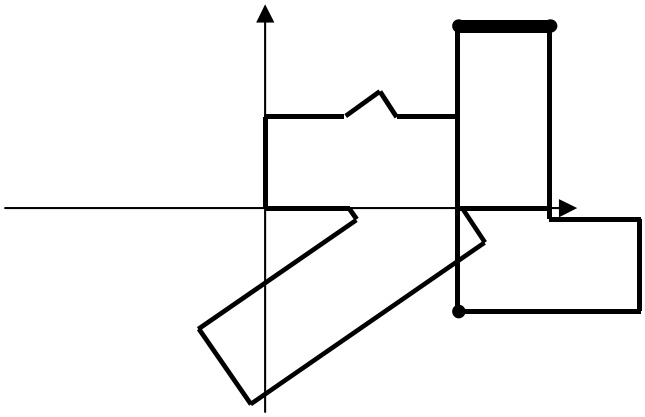


Figura 4.2 – Visão gráfica do casamento da linha F_1 com a linha f_1.

Consideramos uma outra hipótese para F_1. A linha F_1 não pode ter como representante a linha f_2, uma vez que f_2 é maior que F_1 e foi assumido que não há transformação de escala na visualização. Caso F_1 tenha como representante na visualização a linha f_3, então para isto são necessárias uma rotação seguida de uma translação. Graficamente, temos o resultado da Figura 4.3.

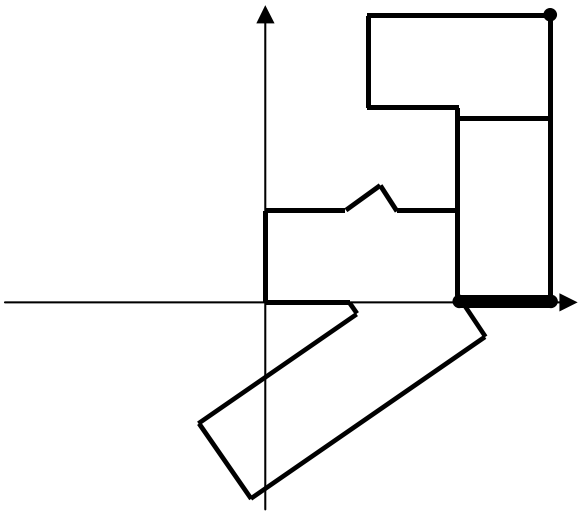


Figura 4.3 – Visão gráfica do casamento da linha F_1 com a linha f_3.

Agora, para o caso da linha f_4 ser a representante da linha F_1, teremos uma rotação e diversas possíveis translações, uma vez que o comprimento da linha f_4 é inferior ao da linha F_1. Graficamente teremos a ilustração da Figura 4.4.

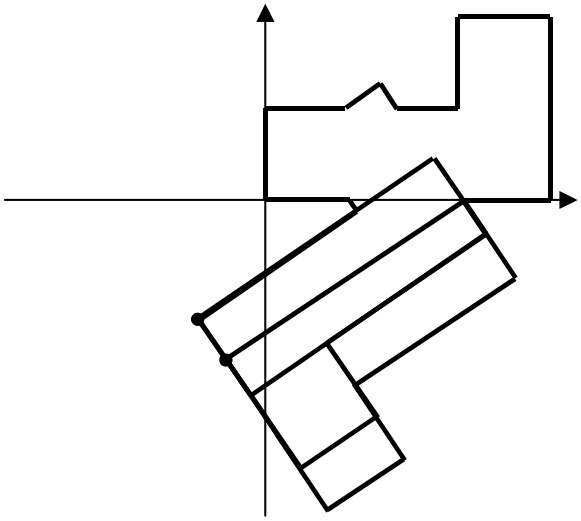


Figura 4.4 – Visão gráfica do casamento da linha F_1 com a linha f_4.

O caso da linha f_5 é análogo ao da linha f_2 – o comprimento da linha f_5 é maior que o da F_1. Para a linha f_6, teremos uma rotação e uma translação, graficamente representadas na Figura 4.5.

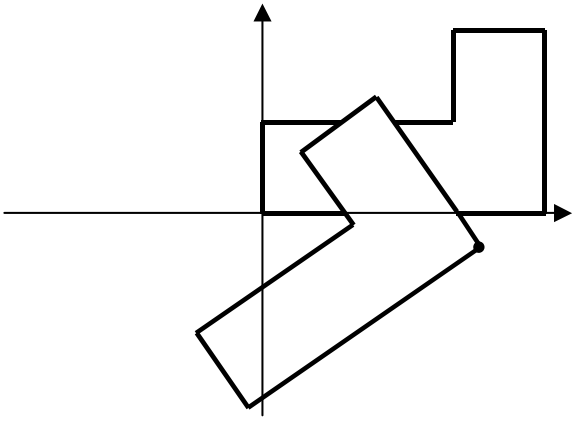


Figura 4.5 – Visão gráfica do casamento da linha F_1 com a linha f_6.

Pode-se observar que todas as linhas do modelo encaixaram-se neste caso exatamente sobre algumas linhas da visualização. Estes passos são repetidos para todas as demais linhas da visualização. Para a linha f_{10} temos novamente que todas as linhas do modelo encaixaram-se exatamente sobre algumas linhas da visualização, conforme visto na Figura 4.6.

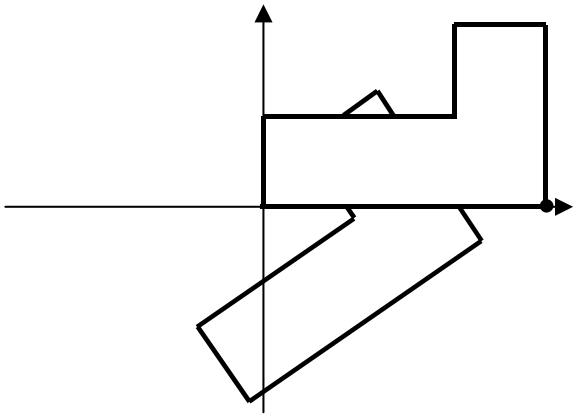


Figura 4.6 – Visão gráfica do casamento da linha F_1 com a linha f_{10}.

Uma vez mapeada linha F_1, passamos para a próxima linha do modelo, a linha F_2, e assim sucessivamente para todas as demais linhas. No final de todas as linhas, teremos o espaço de parâmetros "construído", sendo que os valores de cada ponto indicam quantas vezes foi feita uma transformação de rotação seguida de uma translação para casar linhas do modelo com linhas da visualização. Portanto, a solução é dada pelo ponto do espaço de parâmetros que apresentar o maior valor. Poderá haver mais de uma solução, como podemos notar na Figura 4.5 e na Figura 4.6.

Note que, com este processo, no resultado final obtemos a estimativa de localização do modelo na visualização. Para obter as relações entre as linhas do modelo e as linhas da visualização, basta aplicar a devida transformação representada nos pontos obtidos no espaço dual.

Os problemas com este método de reconhecimento usando a transformada de Hough são os mesmos que ocorrem quando desejamos localizar retas na imagem, como visto na Seção 3.1. O principal problema está relacionado com a necessidade de realizar uma discretização do espaço de parâmetros, o que pode acarretar erros na transformação, como visto no caso de retas.

Se forem realizadas mais transformações, como escala e deformações, principalmente distorções perspectivas existentes em todas as nossas imagens, pois são provenientes de câmeras, a complexidade de utilizar este método aumentará, pois, além de aumentar a dimensão do espaço dual, temos que encontrar todas as possíveis transformações. Se considerarmos o modelo de um campo de futebol dado pela Figura 4.7 e a visualização com uma distorção perspectiva conforme a Figura 4.8, teremos este problema: se fizéssemos o

casamento de F_1 com f_1 apenas com uma rotação e translação sem fazer ao mesmo tempo uma outra transformação que fosse a responsável pela distorção perspectiva, não teríamos nenhum casamento de outro par de linhas modelo-visualização. Faz-se necessário então encontrar uma distorção perspectiva para possibilitar outros casamentos. Neste ponto encontra-se outra dificuldade, relacionada à forma de descrever esta distorção perspectiva para utilizá-la na transformada de Hough.

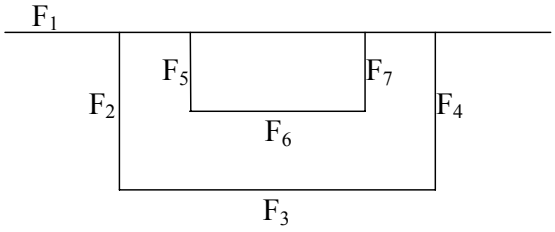


Figura 4.7 – Modelo de um campo de futebol.

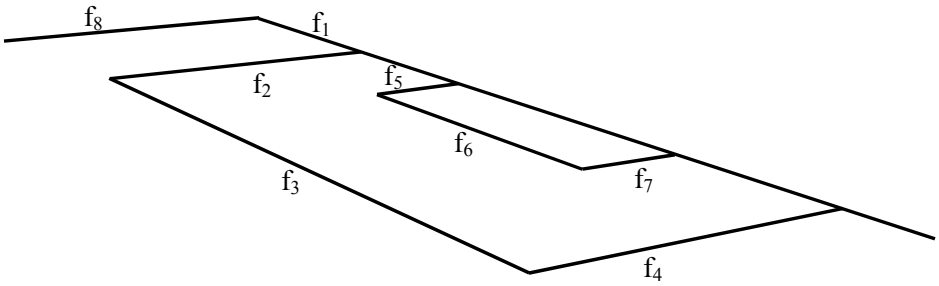


Figura 4.8 – Visualização de um campo de futebol com distorção perspectiva.

Como vemos, este processo é muito ineficiente e difícil de implementar completamente, com diversas transformações ao mesmo tempo.

4.2 Método de Reconhecimento Baseado em Modelos

O método de reconhecimento baseado em modelos [Grimson90] difere conceitualmente do método anterior pelo fato deste trabalhar com interpretação e não com a localização de transformações que associem linhas do modelo a linhas da visualização.

No método de reconhecimento de um conjunto de linhas baseado em modelos, a palavra-chave é *interpretação*. Ele é totalmente baseado em interpretações de ambos os conjuntos de dados – o modelo real e o conjunto de dados de entrada. Com isto, são verificadas as suas equivalências, de acordo com certas interpretações, através da análise de um conjunto de restrições para validar, ou não, uma solução.

Neste método, é utilizada uma estrutura de árvore chamada de "árvore de interpretação", na qual cada folha representa uma possível solução. Em cada nó da árvore é feita uma interpretação para verificar a validade da possível solução do reconhecimento. A profundidade da árvore depende do tamanho do modelo – ela é igual ao número de linhas do modelo real. A Figura 4.9 ilustra os dois primeiros níveis de uma árvore de interpretação. Nesta figura, o modelo real apresenta duas linhas a menos (relativas às linhas laterais de um campo de futebol) que o modelo real utilizado no caso desta tese, como é descrito em uma seção a seguir, apenas para não sobrecarregar a árvore de interpretação da figura.

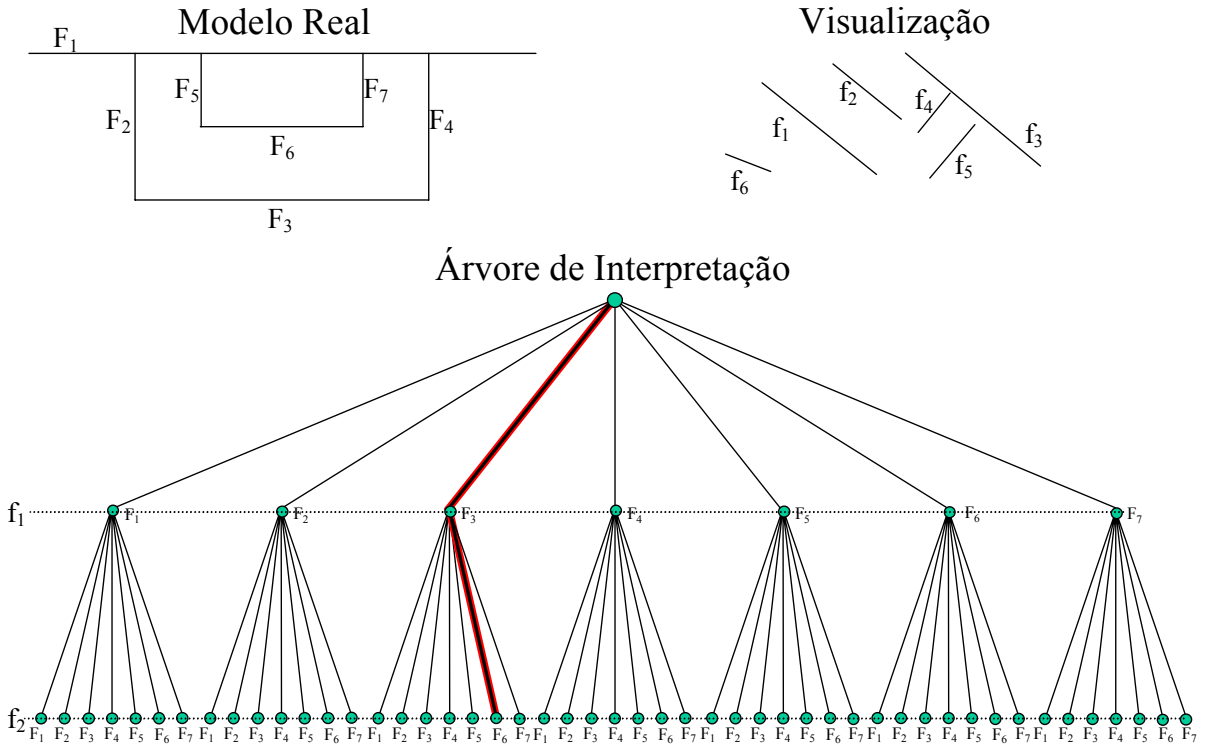


Figura 4.9 – Exemplo de uma árvore de interpretação.

Um exemplo de resultado para a interpretação da visualização em relação ao modelo real ilustrado na Figura 4.9 é:

$$f_1 : F_3, \quad f_2 : F_6, \quad f_3 : F_1, \quad f_4 : F_7, \quad f_5 : F_4, \quad f_6 : \varnothing,$$

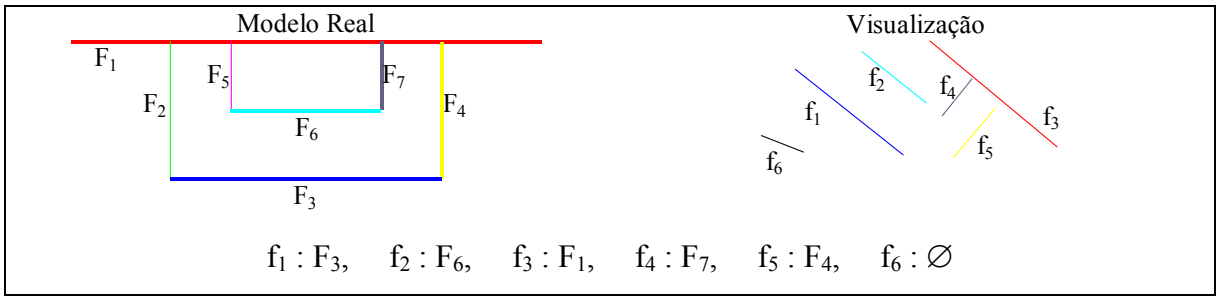
onde $f_x : F_y$ significa que a linha f_x representa a linha F_y e $f_x : \varnothing$ indica que a linha f_x não representa nenhuma linha do modelo real. A profundidade da árvore é igual ao número de linhas na visualização. No exemplo da Figura 4.9, a árvore tem profundidade igual a 6, pois existem 6 linhas no conjunto de visualização.

Nas subseções seguintes são descritas as características deste método, enumerando os pontos positivos e negativos, e é visto como são verificadas as validades das interpretações através de um conjunto de restrições.

4.2.1 Características do Método

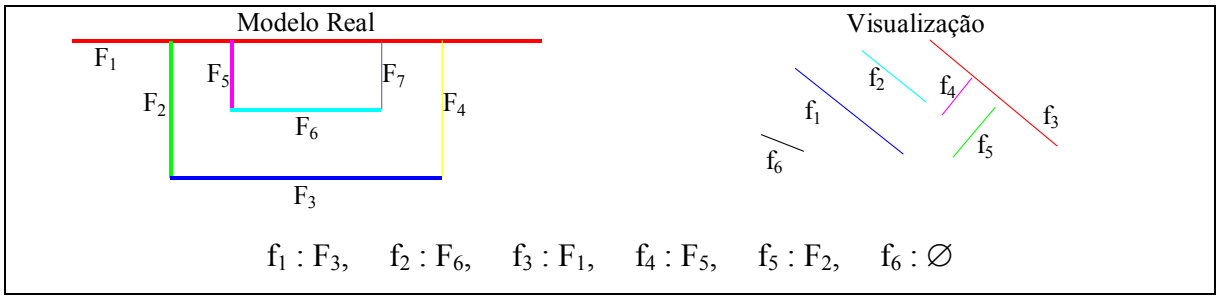
Algumas características importantes em relação a este método, tanto para o caso geral quanto para o caso específico de um modelo de um campo de futebol, são relacionadas abaixo:

- Nem todas as linhas do modelo real têm necessariamente representações na visualização: no exemplo da Figura 4.9, a linha F_2 do modelo não tem representação na visualização;
- Nem todas as linhas da visualização têm necessariamente representações no modelo real: no exemplo da Figura 4.9, a linha f_6 da visualização não tem representação no modelo real. Neste caso, representa-se como $f_6 : \varnothing$;
- Pode surgir mais de uma solução quando:
  - ✓ O modelo é simétrico: o nosso modelo real apresenta uma simetria, acarretando que dada uma solução, como (as linhas mais espessas do modelo indicam as linhas que tem representação na visualização)



$f_1 : F_3, \quad f_2 : F_6, \quad f_3 : F_1, \quad f_4 : F_7, \quad f_5 : F_4, \quad f_6 : \varnothing$

a sua simétrica também seja válida:



$f_1 : F_3, \quad f_2 : F_6, \quad f_3 : F_1, \quad f_4 : F_5, \quad f_5 : F_2, \quad f_6 : \varnothing$

Neste caso, as duas soluções são equivalentes, não fazendo diferença para o nosso caso;

- ✓ A visualização apresenta muitas linhas: no exemplo, se existisse uma linha f_7 paralela à linha f_5 na visualização, teríamos as 4 soluções:

  $f_1 : F_3,\quad f_2 : F_6,\quad f_3 : F_1,\quad f_4 : F_7,\quad f_5 : F_4,\quad f_6 : \varnothing,\quad f_7 : \varnothing,$ (sol. 1)

  $f_1 : F_3,\quad f_2 : F_6,\quad f_3 : F_1,\quad f_4 : F_7,\quad f_5 : \varnothing,\quad f_6 : \varnothing,\quad f_7 : F_4,$ (sol. 2)

  $f_1 : F_3,\quad f_2 : F_6,\quad f_3 : F_1,\quad f_4 : F_5,\quad f_5 : F_2,\quad f_6 : \varnothing,\quad f_7 : \varnothing,$ (sol. 3)

  e

  $f_1 : F_3,\quad f_2 : F_6,\quad f_3 : F_1,\quad f_4 : F_5,\quad f_5 : \varnothing,\quad f_6 : \varnothing,\quad f_7 : F_2.$ (sol. 4)

  Neste caso, as soluções 1 e 3 são idênticas, assim como as soluções 2 e 4, da mesma forma que no caso anterior (simetria). Porém, a diferença entre as soluções 1 e 2 representa uma escala na visualização em apenas uma direção, isto é, representa uma distorção (análogo para as soluções 3 e 4).

- Não existe uma solução possível além da trivial: dependendo do conjunto de interpretações (como é visto adiante), pode não existir uma solução além da trivial, onde todas as linhas da visualização têm representação nula;
- Todos os nós pais de uma possível solução válida também representam uma possível solução válida, na qual as linhas representadas pelos níveis seguintes têm representação nula.

Algumas dessas características podem ser evitadas dependendo do conjunto de restrições impostas ou de critérios de desempate, como é visto na Seção 4.2.2.

A ordem de complexidade desse método e comparações de performance são encontrados em [Fisher94].

4.2.2 Restrições Referentes à Visualização de um Campo de Futebol com Distorção Perspectiva

Quando assistimos a uma partida de futebol, observamos que na grande parte do tempo apenas metade do campo é vista. Portanto, podemos modelar o campo, para o caso desta tese, apenas contendo as linhas de uma metade, sem levar em consideração a linha de meio de campo. A ilustração desse modelo real é visto na Figura 4.10.

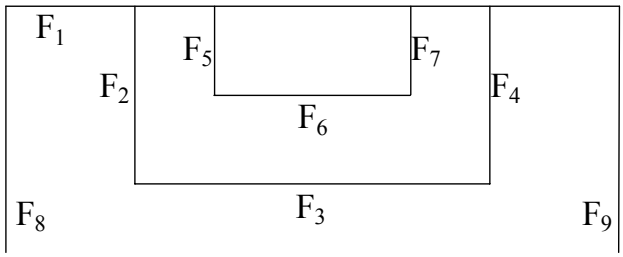


Figura 4.10 – Modelo real de um campo de futebol.

Na nossa modelagem, é permitido que existam linhas da visualização que não tenham representações no modelo real e vice-versa, e as seguintes restrições são impostas:

1. Duas linhas na visualização não podem ter a mesma representação no modelo;
2. Duas linhas são paralelas (ou próximas de paralelas, em virtude da transformação projetiva) na visualização se, e somente se, suas representantes no modelo real também forem paralelas;
3. Todas as linhas devem estar em um mesmo semi-plano determinado pela linha que representa F_1;
4. Todas as linhas devem estar em um mesmo semi-plano determinado pela linha que representa F_8;
5. Todas as linhas devem estar em um mesmo semi-plano determinado pela linha que representa F_9;
6. Todas as linhas devem estar em um mesmo semi-plano determinado pela linha que representa F_2, exceto as representantes de F_1 e F_8;
7. Todas as linhas devem estar em um mesmo semi-plano determinado pela linha que representa F_3, exceto as representantes de F_8 e F_9;
8. Todas as linhas devem estar em um mesmo semi-plano determinado pela linha que representa F_4, exceto as representantes de F_1 e F_9;
9. A linha que representa F_5 deve estar entre as linhas que representam F_1 e F_6;
10. A linha que representa F_5 deve estar entre as linhas que representam F_2 e F_6;
11. A linha que representa F_6 deve estar entre as linhas que representam F_1 e F_3;
12. A linha que representa F_7 deve estar entre as linhas que representam F_1 e F_6;
13. A linha que representa F_7 deve estar entre as linhas que representam F_4 e F_6;
14. O correspondente de F5 não pode cruzar o correspondente de F6 e vice-versa;
15. O correspondente de F7 não pode cruzar o correspondente de F6 e vice-versa.

Note que a restrição 2 acima cita que teremos que considerar segmentos paralelos mesmo quando eles não se devem à transformação projetiva perspectiva. Para isto, devemos suavizar a condição de paralelismo de dois segmentos de reta, $s = \overline{(s_{u1}, s_{v1})(s_{u2}, s_{v2})}$ e $t = \overline{(t_{u1}, t_{v1})(t_{u2}, t_{v2})}$, dada por:

$$\frac{\left|(s_{u2} - s_{u1})(t_{u2} - t_{u1}) + (s_{v2} - s_{v1})(t_{v2} - t_{v1})\right|}{\sqrt{(s_{u2} - s_{u1})^2 + (s_{v2} - s_{v1})^2}\sqrt{(t_{u2} - t_{u1})^2 + (t_{v2} - t_{v1})^2}} = 1 \qquad (4.1)$$

substituindo-a por:

$$\frac{\left|(s_{u2} - s_{u1})(t_{u2} - t_{u1}) + \eta^2(s_{v2} - s_{v1})(t_{v2} - t_{v1})\right|}{\sqrt{(s_{u2} - s_{u1})^2 + \eta^2(s_{v2} - s_{v1})^2}\sqrt{(t_{u2} - t_{u1})^2 + \eta^2(t_{v2} - t_{v1})^2}} \geq \rho \qquad (4.2)$$

onde η é o fator de distorção perspectiva dada na direção $\overrightarrow{ov}$ da imagem e ρ a tolerância de paralelismo, isto é, uma relaxação no critério de paralelismo, conforme a equação (4.1). A existência do fator η na equação (4.2) está relacionada com os pontos de fuga dados pelas direções $\overrightarrow{ox}$ e $\overrightarrow{oy}$ da cena na visualização: o ângulo formado entre elas normalmente é obtuso e para cima, na direção $\overrightarrow{ov}$ da imagem. O fator η aplica uma escala da direção $\overrightarrow{ov}$ nos vetores que indicam as direções de cada linha do campo, com a função de diminuir a abertura desse ângulo. Quando o ponto de fuga não é um ponto impróprio, para encontrar este ângulo basta subtrair os pontos de fuga pelo centro da imagem. Isto pode ser observado na Figura 4.11, que exemplifica um esquema de visualização comum transmitido pelas emissoras de televisão. Nesta figura, o círculo vermelho indica o ponto de fuga da direção $\overrightarrow{ox}$, o círculo azul é referente à direção $\overrightarrow{oy}$ e o círculo preto indica o centro da imagem.

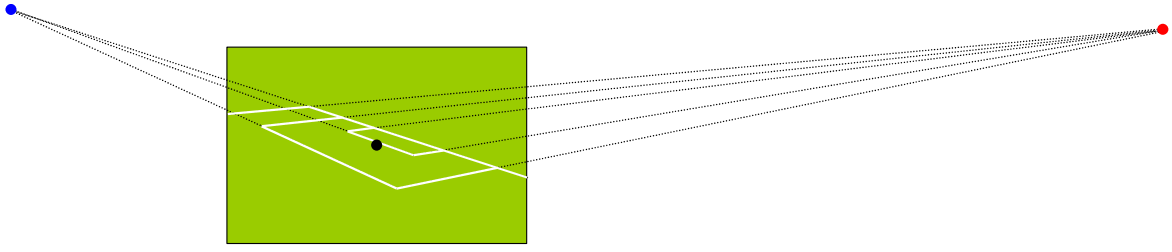


Figura 4.11 – Pontos de fuga da visualização de um campo de futebol.

Quando este ângulo é agudo, a distorção é muito grande e não é comum termos uma visão assim, como ilustrado na Figura 4.12. Os círculos presentes nesta figura têm os mesmos significados que na Figura 4.11.

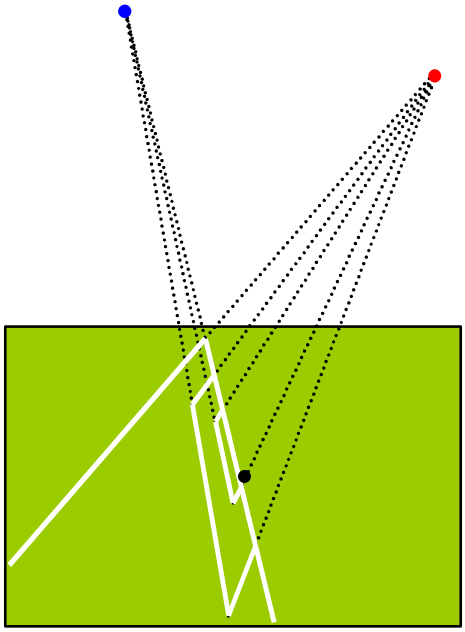


Figura 4.12 – Visualização com grande distorção perspectiva.

Na aplicação desenvolvida nesta tese para testar os algoritmos, os valores dessas duas constantes foram 2 e 0,81, respectivamente. O primeiro valor, $\eta = 2$, suaviza a abertura do ângulo, como vemos na Figura 4.13. Nesta mesma figura observamos que linhas do campo, que na verdade são paralelas, na visualização apresentam ângulos que dificilmente são maiores de 36 graus, cujo co-seno é aproximadamente 0,81, valor dado para ρ na equação (4.2).

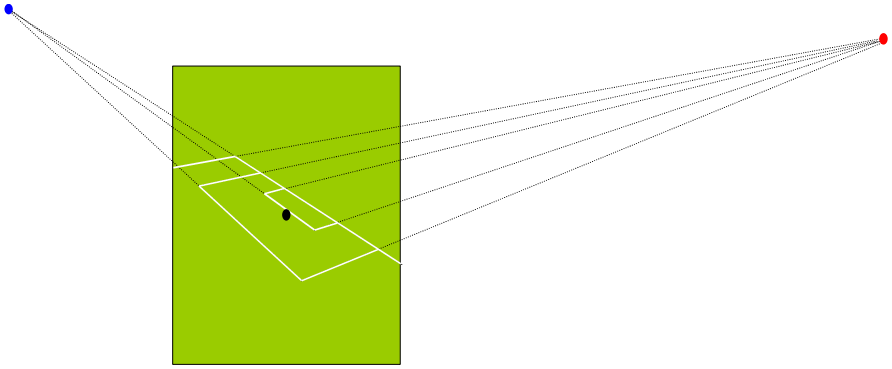


Figura 4.13 –Figura 4.11 com escala 2 na direção $\overrightarrow{ov}$.

Para que uma solução seja válida para o algoritmo de calibração automática de câmera proposto, é necessário que determinadas linhas do modelo real tenham representações na visualização. Três conjuntos mínimos de linhas do modelo real são $\{F_1, F_2, F_3, F_5, F_6\}$, $\{F_1, F_3, F_4, F_6, F_7\}$ e $\{F_1, F_2, F_3, F_4, F_6\}$. Os dois primeiros representam na realidade a mesma

solução, com a diferença apenas de um espelhamento.

Além da lista de restrições, é aplicado ainda um critério de desempate, eliminando de início a solução trivial – na qual nenhuma linha da visualização tenha representação no modelo real. Alguns exemplos de critérios de desempate são:

- vence a solução com o maior número de linhas que tenham representações;
- vence a solução que tenha a maior soma dos comprimentos das linhas que tenham representações;
- vence a solução que apresente a maior linha com representação da linha F_0 do modelo real;
- combinação dos exemplos acima.

Em relação a Figura 2.1, através do método proposto na Seção 3.2, temos a lista de segmentos de retas longos descrita na Tabela 3.1. Duas soluções de reconhecimento equivalentes resultados do método proposto aqui para os segmentos dessa tabela estão ilustrados na Figura 4.14 e na Figura 4.15.

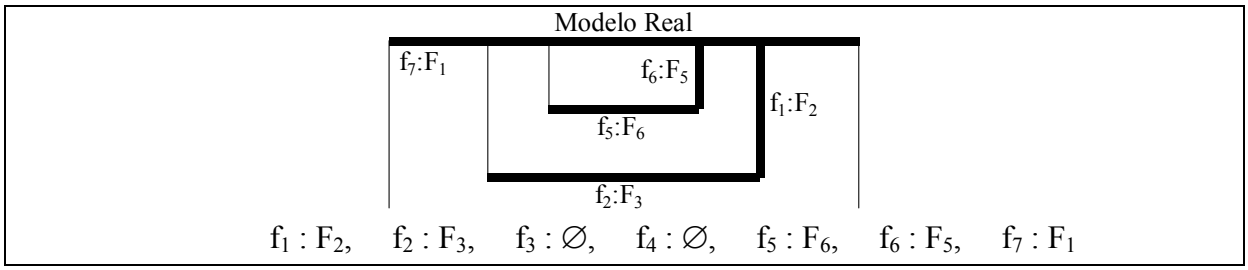


Figura 4.14 – Reconhecimento da imagem da Figura 3.18, segundo modelo da Figura 4.10.

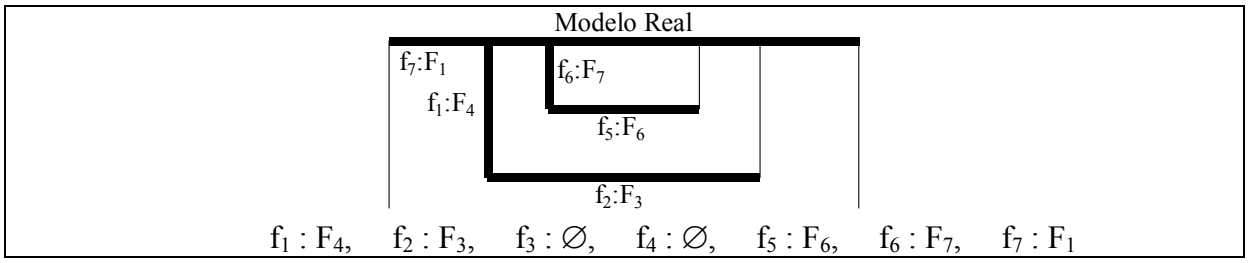


Figura 4.15 – Solução equivalente à ilustrada na Figura 4.14.

As linhas mais espessas da Figura 4.14 e da Figura 4.15 indicam que tem representatividade na visualização. Verificamos que o resultado é satisfatório, observando a Figura 3.19. Visualmente, referente à Figura 3.19, a primeira solução parece mais adequada, embora não exista nada contra a segunda solução.

4.2.2.1 Exemplo de uma Solução Inválida

Durante a construção da árvore de interpretação, analisamos se o conjunto de restrições continua válido, isto é, se todas as restrições ainda são respeitadas para o modelo. Abaixo damos um exemplo de uma solução parcial que invalida o conjunto de restrições.

Suponhamos que a extração de segmentos de retas longos de uma imagem de um campo de futebol tenha como resultado os segmentos da Figura 4.16. Então a solução parcial referente ao nó

$$f_1 : F_6, \quad f_2 : F_3, \quad f_3 : F_1$$

é inválida pois contradiz a restrição 11 (a linha que representa F_6 deve estar entre as linhas que representam F_1 e F_3). Com isto, não precisamos percorrer os nós filhos desta árvore, pois todos eles vão conter esta solução parcial inválida. Este fato pode eliminar de uma só vez um número grande de soluções inválidas encontradas nas folhas da árvore que estão abaixo desse nó.

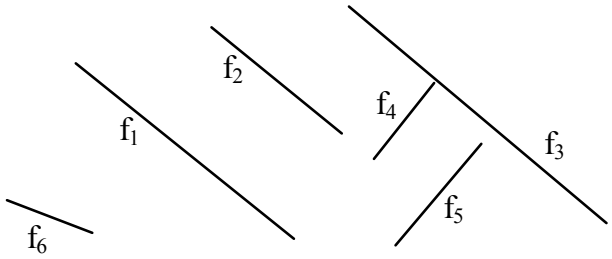


Figura 4.16 – Exemplo de visualização das linhas do campo.

4.2.3 Generalização das Restrições

Na Seção 4.2.2, foi apresentado um conjunto de 15 restrições que caracterizam o conjunto das linhas de uma metade de um campo de futebol. Uma característica desse conjunto é o fato das linhas que o compõem poderem ser divididas em dois grupos: um grupo formado pelas linhas paralelas à linha de fundo (linha do campo que contém um gol) e outro formado pelas demais (ortogonais à linha de fundo).

Para modelos compostos por segmentos paralelos a duas direções mutuamente ortogonais, é possível gerar, automaticamente, utilizando apenas a descrição geométrica de cada segmento, um conjunto de restrições que permite interpretar uma imagem do modelo, utilizando o método da subseção anterior.

As restrições genéricas geradas para um modelo composto de segmentos de retas que se apresentam em apenas duas direções, perpendiculares ou ortogonais, podem ser enumeradas da seguinte forma:

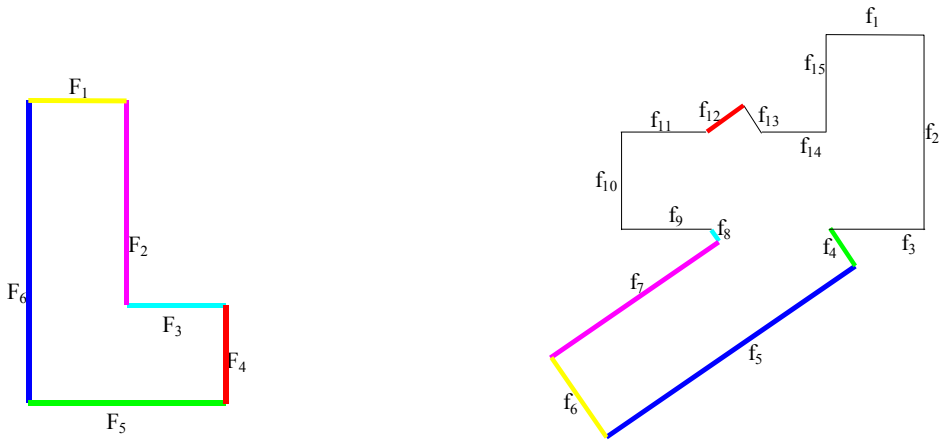
1. Duas linhas na visualização não podem ter a mesma representação no modelo;
2. Se duas linhas no modelo são paralelas, então suas correspondentes na visualização também devem ser paralelas;
3. Se duas linhas no modelo são ortogonais, então suas correspondentes na visualização também devem ser ortogonais;
4. Se uma linha *i* está entre duas linhas *j* e *k* no modelo, então a representante de *i* deve estar entre as representantes de *j* e *k* na visualização;
5. Se duas linhas se cruzam na visualização, então suas representantes no modelo também devem se cruzar.

As restrições acima estão relacionadas com o cuidado de manter a ordem e as direções das linhas do modelo de acordo com suas representantes na visualização.

A partir dessa generalização, a solução encontrada pelo método para o modelo e visualização ilustrados na Figura 4.1(a) e Figura 4.1 (b), respectivamente é:

$f_1 : \varnothing$, $f_2 : \varnothing$, $f_3 : \varnothing$, $f_4 : F_5$, $f_5 : F_6$, $f_6 : F_1$, $f_7 : F_2$, $f_8 : F_3$, $f_9 : \varnothing$, $f_{10} : \varnothing$, $f_{11} : \varnothing$, $f_{12} : F_4$, $f_{13} : \varnothing$, $f_{14} : \varnothing$, $f_{15} : \varnothing$.

Visualmente, esta solução é vista na Figura 4.17, onde as linhas de mesma cor no modelo e na visualização indicam a representatividade. As linha em preto na visualização indicam que não há representatividade no modelo.



(a) Modelo (b) Visualização

Figura 4.17 – Resultado visual do método de reconhecimento para a Figura 4.1.

4.3 Conclusões

Embora a transformada de Hough solucione o problema de reconhecer e localizar um dado padrão em uma imagem, quando esta é proveniente de uma câmera, incorporando, entre outras dificuldades, distorção perspectiva, a utilização desse método não é muito eficiente. Outros problemas com a utilização dessa transformada são semelhantes aos encontrados quando ela é aplicada à técnica de extração de retas, como lentidão e falta de precisão.

Devido a estes problemas, buscou-se outro método, baseado em uma árvore de interpretação. É um método mais fácil de implementar e como as restrições, que auxiliam na interpretação entre a visualização e o modelo real, são escritas livremente pela pessoa que irá implementar o método, torna-se possível resolver problemas que com a utilização da transformada de Hough teriam soluções complexas.

Para o caso do objeto desta tese, foi elaborado um conjunto pequeno de 5 restrições que possibilita a utilização do método de reconhecimento baseado na árvore de interpretação para que um modelo real possa ser abstrato e genérico, apenas com a restrição de conter segmentos de retas paralelos ou perpendiculares entre si.

A grande vantagem do método baseado na árvore de interpretação é uma maior independência para lidar com as transformações que um modelo pode sofrer na visualização. Uma desvantagem é a possibilidade do método retornar mais de uma solução e, se o critério de desempate não for bem elaborado, surgir um resultado não esperado.

Apesar da árvore poder possuir um grande número de nós, quando completa, muitas vezes uma restrição elimina (poda) grande parte dos nós, como é o caso das restrições 2 e 3 listadas na seção anterior. O tempo de resposta do método foi muito bom, não prejudicando o processamento do algoritmo como um todo em tempo real.

A equação (4.2) descreve bem o comportamento entre as linhas, verificando o paralelismo entre elas mesmo quando está embutida a distorção perspectiva na imagem. O parâmetro η resolve a distorção maior existente na direção $\overrightarrow{ov}$ da imagem, enquanto o parâmetro ρ relaxa o critério de paralelismo, que seria igual a 1. Se for apenas usada a equação (4.1) adicionando a relaxação com o parâmetro ρ, é difícil avaliar o paralelismo entre dois segmentos, pois, devido à perspectiva, na mesma visualização duas linhas paralelas no modelo podem aparecer na imagem possuindo uma interseção (ponto de fuga próprio),

exemplificada na Figura 4.18 pelas linhas f_1 e f_3. A linha f_2 no modelo é perpendicular às linhas f_1 e f_3.

Como podemos ver na Figura 4.18, existe um triângulo (desenhado na cor vermelha) a partir das retas suportes das linhas f_1, f_2 e f_3 que pode ser isósceles, isto é, os lados referentes a f_1 e f_2 podem ter o mesmo comprimento. Isto significa que, na visualização, o ângulo formado pelos lados f_1 e f_3 é igual ao formado pelos lados f_2 e f_3, dificultando determinar na visualização quem é paralelo ou perpendicular a quem. Portanto, deve-se prestar atenção ao determinar o parâmetro η. Se a linha f_3 fosse paralela ao eixo $\overrightarrow{ou}$ da imagem, não teríamos como resolver este problema.

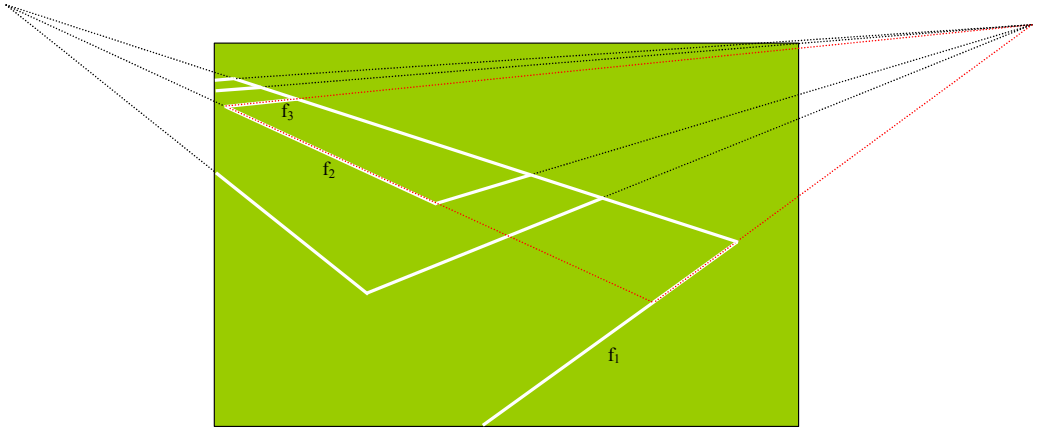


Figura 4.18 – Visão perspectiva: caso de ângulos iguais para pares de linhas diferentes.

Capítulo 5

5 Câmeras e Homografia

Este capítulo aborda técnicas para encontrar transformações que levam pontos do mundo tridimensional para o plano da imagem. São apresentados algoritmos para determinar uma homografia (transformação projetiva planar), na qual é encontrado um mapeamento de 2D para 2D, de um plano contido no mundo real para o plano da imagem. Também são abordados dois métodos relativos à calibração de câmeras, o Método de Tsai e o do Juiz Virtual, através dos quais são encontradas transformações projetivas de 3D para 2D. Todos esses métodos são baseados no modelo de câmera fotográfica tipo "*pinhole*" de projeção cônica, que é discutido na primeira seção deste capítulo, fazendo uma referência a homografias quando temos modelos coplanares.

5.1 Modelo de Câmera "Pinhole"

O modelo de câmera "*pinhole*" é bem simples. O seu funcionamento é baseado nos raios que partem de objetos no mundo tridimensional e passam por um orifício, formando uma imagem invertida em um anteparo. Este esquema está ilustrado na Figura 5.1, onde o orifício é indicado como centro de projeção. Uma câmera desse tipo é bem fácil de ser construída, como podemos ver na Figura 5.2. A câmera ilustrada nessa figura é feita a partir de uma lata com um pequeno furo (orifício) por onde passam os raios, formando a imagem no fundo da lata, onde pode ser posto um filme para capturá-la. Na frente do orifício coloca-se um papel, que só é retirado no momento em que queremos capturar a imagem.

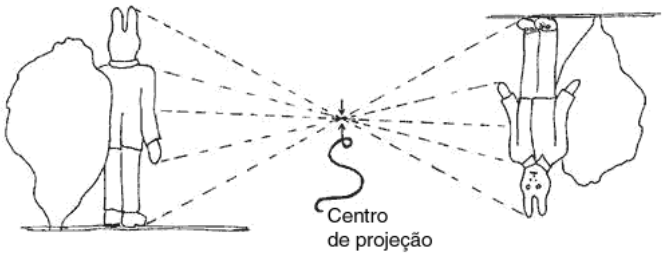


Figura 5.1 – Projeção através de um centro (ponto).

Figura 5.2 – Câmera do tipo "*pinhole*" feita de lata.

Em computação gráfica, este modelo é modificado, passando a imagem projetada a se formar entre o objeto tridimensional e o centro de projeção. A imagem é formada então em um plano a uma distância da câmera chamada de distância focal, denotada por *f*.

Consideremos um sistema de coordenadas tridimensional *OXYZ*, com a origem no centro óptico da câmera, o eixo *Z* perpendicular ao plano de projeção onde a imagem é formada e os eixos *X* e *Y* paralelos aos lados da imagem, como mostrado na Figura 5.3. A projeção desse sistema no plano da imagem induz um sistema de coordenadas $C\tilde{u}\tilde{v}$ nesse plano.

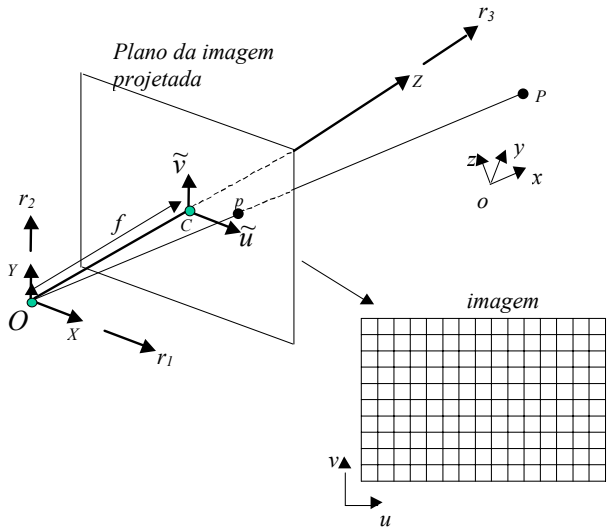


Figura 5.3 – Modelo de câmera "*pinhole*".

Com relação a esses sistemas de coordenadas, a projeção perspectiva (ou cônica) $p = (\tilde{u}, \tilde{v})$ de um ponto $P = (X, Y, Z)$ do espaço no sistema de coordenadas *CXYZ* é dada por

$$(\tilde{u}, \tilde{v}) = \left(f\frac{X}{Z}, f\frac{Y}{Z} \right) \tag{5.1}$$

onde $(\tilde{u}, \tilde{v}) = (u - u_0, v - v_0)$ e $C = (u_0, v_0)$ é o centro da imagem.

Mas, normalmente, um ponto no espaço tridimensional é descrito por outro sistema de coordenadas – *oxyz*. Para utilizar a equação (5.1) para um ponto no sistema *oxyz*, primeiro deve-se realizar uma transformação que mapeie este ponto para o sistema da câmera. Esta transformação pode ser obtida pela composição de uma rotação que alinhe os eixos seguida de uma translação que compatibilize as origens. Esta transformação é descrita pela seguinte matriz 3 × 4:

$$[R \mid T] = \begin{bmatrix} r_1 & t_x \\ r_2 & t_y \\ r_3 & t_z \end{bmatrix} = \begin{bmatrix} r_{1x} & r_{1y} & r_{1z} & t_x \\ r_{2x} & r_{2y} & r_{2z} & t_y \\ r_{3x} & r_{3y} & r_{3z} & t_z \end{bmatrix} \tag{5.2}$$

onde r_1, r_2 e r_3 definem a direção e inclinação da câmera, conforme ilustrado na Figura 5.3, formando uma base ortonormal. Convém observar que o ponto $T = (t_x, t_y, t_z)$ representa a origem do sistema *oxyz* escrita em coordenadas do sistema da câmera.

Logo, compatibilizando as equações (5.1) e (5.2), temos que um ponto do mundo tridimensional (x, y, z) projeta-se no ponto $(\tilde{u}, \tilde{v})$ segundo a seguinte equação:

$$\begin{bmatrix} \tilde{u}s \\ \tilde{v}s \\ s \end{bmatrix} = \begin{bmatrix} fr_1 & ft_x \\ fr_2 & ft_y \\ r_3 & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{5.3}$$

Ou seja:

$$u - u_0 = f\frac{r_{1x}x + r_{1y}y + r_{1z}z + t_x}{r_{3x}x + r_{3y}y + r_{3z}z + t_z} \tag{5.4}$$

e

$$v - v_0 = f\frac{r_{2x}x + r_{2y}y + r_{2z}z + t_y}{r_{3x}x + r_{3y}y + r_{3z}z + t_z} \tag{5.5}$$

As equações (5.4) e (5.5) podem ser reescritas na forma matricial como apresentado

em (5.6), utilizando coordenadas homogêneas, onde Q é a transformação de câmera.

$$\begin{bmatrix} us \\ vs \\ s \end{bmatrix} = Q \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} q_1 & q_{14} \\ q_2 & q_{24} \\ q_3 & q_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} fr_1 + u_0 r_3 & ft_x + u_0 t_z \\ fr_2 + v_0 r_3 & ft_y + v_0 t_z \\ r_3 & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad (5.6)$$

Uma observação a ser feita é que a escala existente entre a imagem projetada e a imagem final, conforme ilustra a Figura 5.3, está embutida na distância focal f.

Estas formulações são referentes a uma câmera do tipo "*pinhole*". Porém, se modelarmos a projeção apenas por uma transformação projetiva genérica, sem as restrições desse tipo de câmera, essa transformação seria dada apenas por uma matriz 3×4. Isto é, a transformação seria dada apenas por

$$\begin{bmatrix} us \\ vs \\ s \end{bmatrix} = Q \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad (5.7)$$

onde os coeficientes q_{ij} não teriam as restrições implícitas na equação (5.6).

Em muitos casos, os pontos do mundo tridimensional são coplanares e têm z igual a zero. Isto significa que a transformação da equação (5.7) pode ser reescrita como:

$$\begin{bmatrix} us \\ vs \\ s \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (5.8)$$

A transformação H dada na equação (5.8) é uma transformação projetiva planar, chamada também de homografia.

Nas seções seguintes, são vistos métodos para encontrar homografias e os parâmetros da câmera que formam a matriz dada na equação (5.6). Isto é, dado um conjunto de pares de pontos da forma (P, p), onde P é um ponto do espaço tridimensional e p um ponto da imagem, deve-se encontrar uma homografia, ou uma câmera, de forma que p seja a imagem de P através da homografia, ou da câmera. Os métodos para encontrar os parâmetros da câmera são chamados de *métodos de calibração de câmeras*. O próximo capítulo explicita como obter esses pares (P_k, p_k) a partir do reconhecimento das linhas da imagem do modelo e de suas interseções.

5.2 Estimação de Homografias

Em muitas aplicações, há apenas a necessidade de encontrar uma transformação que mapeie um plano no espaço da cena para o plano da imagem. Este é o caso, por exemplo, de restringir-se a atenção à imagem de um campo de futebol, ignorando-se os objetos situados fora do plano do gramado. Esta é uma transformação projetiva (ou homografia) e é convenientemente descrita utilizando coordenadas homogêneas através da equação

$$H\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} us \\ vs \\ ts \end{bmatrix} \tag{5.9}$$

onde $P = (x, y, 0, w)$ é um ponto do mundo da cena e $p = (u, v, t)$ um ponto no plano da imagem (utilizando coordenadas homogêneas).

Nesta tese, os pontos $P = (x, y, 0, w)$, usados para determinar a transformação da equação (5.1), são calculados a partir da interseção das linhas do modelo, sendo portanto coplanares.

O problema abordado nesta seção consiste em encontrar os valores para h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32} e h_{33}, referentes à homografia H, e também os valores s dos pontos do plano da imagem. Pode-se notar que, se todas estas incógnitas forem multiplicadas por um mesmo valor, a transformação homogênea é a mesma. Além disso, a solução trivial nula satisfaz o problema.

Para evitar a solução trivial nula, escolhemos um dos pares de pontos próprios (P_k, p_k), fixamos a coordenada t_i de p_k como sendo igual a 1 e reordenamos o conjunto de pares de pontos de forma que o ponto escolhido seja o último.

Com isto, podemos reescrever a equação (5.9) na seguinte forma:

$$\begin{aligned} x_k h_{11} + y_k h_{12} + w_k h_{13} - u_k s_k &= 0 \\ x_k h_{21} + y_k h_{22} + w_k h_{23} - v_k s_k &= 0 \\ x_k h_{31} + y_k h_{32} + w_k h_{33} - t_k s_k &= 0 \end{aligned} \tag{5.10}$$

para $k = 1$ até n-1, e

$$\begin{aligned} x_n h_{11} + y_n h_{12} + w_n h_{13} &= u_n \\ x_n h_{21} + y_n h_{22} + w_n h_{23} &= v_n \\ x_n h_{31} + y_n h_{32} + w_n h_{33} &= 1 \end{aligned} \tag{5.11}$$

Reescrevendo estas equações na forma de produto de matrizes, obtemos:

$$\begin{bmatrix} x_1 & y_1 & w_1 & 0 & 0 & 0 & 0 & 0 & 0 & -u_1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & x_1 & y_1 & w_1 & 0 & 0 & 0 & -v_1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x_1 & y_1 & w_1 & -t_1 & 0 & \cdots & 0 \\ x_2 & y_2 & w_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -u_2 & \cdots & 0 \\ 0 & 0 & 0 & x_2 & y_2 & w_2 & 0 & 0 & 0 & 0 & -v_2 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x_2 & y_2 & w_2 & 0 & -t_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n-1} & y_{n-1} & w_{n-1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & -u_{n-1} \\ 0 & 0 & 0 & x_{n-1} & y_{n-1} & w_{n-1} & 0 & 0 & 0 & 0 & 0 & \cdots & -v_{n-1} \\ 0 & 0 & 0 & 0 & 0 & 0 & x_{n-1} & y_{n-1} & w_{n-1} & 0 & 0 & \cdots & -t_{n-1} \\ x_n & y_n & w_n & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & x_n & y_n & w_n & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x_n & y_n & w_n & 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \\ s_1 \\ s_2 \\ \vdots \\ s_{n-1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ u_n \\ v_n \\ 1 \end{bmatrix} \tag{5.12}$$

Observamos que temos um sistema de 3n equações e 9+(n-1) incógnitas para resolver, onde n é o número de pares de pontos (P_k, p_k). Com n = 4, pode-se resolver o sistema diretamente, utilizando o método de Gauss. Geralmente, n é maior que 4, portanto o sistema tem mais equações do que incógnitas. Isto nos leva a utilizar o método de mínimos quadrados, aplicando as equações normais ou decomposição em valores singulares [Carvalho+99].

Outra maneira de formular o problema, baseada apenas em pontos próprios, consiste em determinar h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32} e h_{33}, que satisfazem

$$u_k = \frac{h_{11}x_k + h_{12}y_k + h_{13}w_k}{h_{31}x_k + h_{32}y_k + h_{33}w_k} \tag{5.13}$$

e

$$v_k = \frac{h_{21}x_k + h_{22}y_k + h_{23}w_k}{h_{31}x_k + h_{32}y_k + h_{33}w_k} \tag{5.14}$$

para todos os pares de pontos (P_k, p_k), k = 1, …, n.

Isto se faz minimizando o erro quadrático

$$\sum_{k=1}^{n} \left(u_k - u'_k\right)^2 + \left(v_k - v'_k\right)^2 \tag{5.15}$$

onde $\left(u'_k, v'_k\right)$ é o resultado obtido pela transformação encontrada.

O problema aqui está relacionado com a não linearidade das equações (5.13) e (5.14). Porém, se em vez de utilizarem essas equações, forem usadas

$$\begin{aligned} h_{11}x_k + h_{12}y_k + h_{13}w_k - u_k\left(h_{31}x_k + h_{32}y_k + h_{33}w_k\right) = 0 \\ h_{21}x_k + h_{22}y_k + h_{23}w_k - v_k\left(h_{31}x_k + h_{32}y_k + h_{33}w_k\right) = 0 \end{aligned} \tag{5.16}$$

para todo $k = 1, \ldots, n$, teríamos um problema linear.

Para eliminar a solução trivial $H = 0$, adotamos a seguinte restrição:

$$h_{31}\bar{x} + h_{32}\bar{y} + h_{33} = 1 \tag{5.17}$$

onde $(\bar{x}, \bar{y}, 0, 1)$ é o centróide dos pontos próprios P_k. A idéia por trás dessa escolha é dar, ao denominador da transformação projetiva planar de um ponto "típico" da cena, valor igual a 1. Desde que todos os pontos não estejam muito distantes do centróide, pode-se esperar que o erro médio ocorrido quando aplicada a equação (5.16) não seja muito grande.

A solução desse problema se dá por resolver

Minimizar $\| \boldsymbol{Mt} \|$

sujeito a $\boldsymbol{m}^{\mathrm{T}}\boldsymbol{t} = 1$,

onde

$$t^T = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32} & h_{33} \end{bmatrix}, \tag{5.18}$$

$$m = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & \bar{x} & \bar{y} & 1 \end{bmatrix} \tag{5.19}$$

e M é a representação matricial da transformação da equação (5.16), dada por:

$$M = \begin{bmatrix} x_1 & y_1 & w_1 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 & -u_1 \\ x_2 & y_2 & w_2 & 0 & 0 & 0 & -u_2x_2 & -u_2y_2 & -u_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & w_n & 0 & 0 & 0 & -u_nx_n & -u_ny_n & -u_n \\ 0 & 0 & 0 & x_1 & y_1 & w_1 & -v_1x_1 & -v_1y_1 & -v_1 \\ 0 & 0 & 0 & x_2 & y_2 & w_2 & -v_2x_2 & -v_2y_2 & -v_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & x_n & y_n & w_n & -v_nx_n & -v_ny_n & -v_v \end{bmatrix} \tag{5.20}$$

A dimensão da matriz M é $2n \times 9$.

Para resolver o problema acima, é usado nesta tese o método de multiplicadores de Lagrange, conduzindo ao seguinte sistema linear:

$$\begin{cases} M^T Mt - \lambda m = 0 \\ m^T t = 1 \end{cases} \tag{5.21}$$

onde λ é o multiplicador de Lagrange associado à restrição linear.

Uma vantagem dessa formulação do problema em relação à anterior é que o tamanho do sistema é bem menor, tornando-o mais rápido de ser resolvido, e na prática apresenta melhores resultados. Apresentamos aqui a primeira formulação por ser mais intuitiva e possibilitar a utilização de pontos impróprios.

5.3 Calibração de Câmeras

Nos casos em que desejamos tratar elementos não situados em um único plano, é necessário determinar a câmera que gerou a imagem. Recuperando a câmera, podemos extrair diversas informações que sem ela não seria possível e até mesmo inserir novos objetos na cena, respeitando, entre diversas características, a oclusão.

Uma câmera é composta, ou pode ser modelada, por parâmetros intrínsecos, compostos pela geometria interna e características ópticas, e extrínsecos, dados pela sua localização, direção de visualização e inclinação. Todos estes parâmetros estão embutidos na equação (5.6). Esta seção apresenta dois métodos de calibração de câmeras: um clássico e conhecido na literatura, chamado Método de Tsai [Tsai86], e outro utilizado pelo sistema Juiz Virtual [JuizVirtual].

Calibrar uma câmera consiste em, dada uma amostra de n pontos $P_i = (x_i, y_i, z_i)$ no espaço 3D e suas projeções correspondentes $p_i = (u_i, v_i)$ na imagem, encontrar uma transformação T que mapeie cada ponto P_i no ponto correspondente p_i. Em outras palavras, calibração de câmera, no contexto de visão computacional, consiste no processo de determinar os parâmetros intrínsecos e extrínsecos da câmera. Uma vez recuperados tais parâmetros, é possível realizar uma modelagem da cena (*image-based modeling*) [Carvalho+98].

5.3.1 Método de Tsai

Esta seção, descreve o Método de Tsai para encontrar os parâmetros da câmera apresentados na Seção 5.1. Isto é, apesar do modelo de câmera empregado pelo Método de Tsai possuir outros parâmetros, num total de 11 (5 intrínsecos e 6 extrínsecos), aqui vamos encontrar apenas os parâmetros da câmera da equação (5.6), com exceção do centro da imagem $C = (u_0, v_0)$, que é fixo. Não é levada em consideração a distorção radial de lente,

por exemplo. O método apresentado aqui é apenas para a versão coplanar, isto é, o método que encontra os parâmetros para pontos do mundo tridimensional que tem $z = 0$.

A partir das equações (5.4) e (5.5), assumindo $z = 0$ para todos os pontos, temos:

$$u - u_0 = \tilde{u} = f \frac{r_{1x}x + r_{1y}y + t_x}{r_{3x}x + r_{3y}y + t_z} \tag{5.22}$$

e

$$v - v_0 = \tilde{v} = f \frac{r_{2x}x + r_{2y}y + t_y}{r_{3x}x + r_{3y}y + t_z} \tag{5.23}$$

Dividindo (5.22) por (5.23) e em seguida o numerador e o denominador da razão da direita por t_y, temos:

$$\frac{\tilde{u}_i}{\tilde{v}_i} = \frac{\frac{r_{1x}}{t_y}x_i + \frac{r_{1y}}{t_y}y_i + \frac{t_x}{t_y}}{\frac{r_{2x}}{t_y}x_i + \frac{r_{2y}}{t_y}y_i + 1} \tag{5.24}$$

ou seja,

$$\frac{r_{1x}}{t_y}x_i\tilde{v}_i + \frac{r_{1y}}{t_y}y_i\tilde{v}_i - \frac{r_{2x}}{t_y}x_i\tilde{u}_i - \frac{r_{2y}}{t_y}y_i\tilde{u}_i + \frac{t_x}{t_y}\tilde{v}_i = \tilde{u}_i \tag{5.25}$$

Obtemos com isto um sistema linear $Au = b$, onde A é uma matriz $n \times 5$ e cada linha A_i é dada por $\left(x_i\tilde{v}_i, y_i\tilde{v}_i, -x_i\tilde{u}_i, -y_i\tilde{u}_i, \tilde{v}_i\right)$, u sendo um vetor dado por

$$\left(U_1, U_2, U_3, U_4, U_5\right) = \left(\frac{r_{1x}}{t_y}, \frac{r_{1y}}{t_y}, \frac{r_{2x}}{t_y}, \frac{r_{2y}}{t_y}, \frac{t_x}{t_y}\right) \tag{5.26}$$

e cada elemento do vetor b sendo dado por $\tilde{u}_i$.

Como r_1, r_2 e r_3 são ortonormais, e designando $\alpha = \frac{r_{1z}}{t_y}$ e $\beta = \frac{r_{2z}}{t_y}$, temos que

$$\begin{cases} \alpha\beta = -U_1U_3 - U_2U_4 \\ \alpha^2 + U_1^2 + U_2^2 = \beta^2 + U_3^2 + U_4^2 \end{cases} \tag{5.27}$$

Resolvendo o sistema acima, calcula-se t_y a partir de

$$t_y^2 = \frac{U - \sqrt{U^2 - 4\left(U_1U_4 - U_2U_3\right)^2}}{2\left(U_1U_4 - U_2U_3\right)^2} \tag{5.28}$$

onde $U = U_1^2 + U_2^2 + U_3^2 + U_4^2$.

Portanto, com estes valores já definidos e com a equação (5.18), encontramos os valores de r_{1x}, r_{1y}, r_{2x}, r_{2y} e t_x. Pelo fato dos vetores r_1 e r_2 terem normas iguais a 1, encontram-se também os valores de r_{1z} e r_{2z}, e conseqüentemente calcula-se r_3, pois r_1, r_2 e r_3 são ortonormais. Finalmente, usando os valores já encontrados e as equações (5.22) e (5.23), calculamos f e t_z. Pode haver a necessidade de ajustar sinais que inicialmente assumimos como positivos, como, por exemplo, t_y.

Uma otimização pode ser feita reajustando estes parâmetros, minimizando o erro cometido entre os pontos dados da imagem e os pontos encontrados pela câmera. O resultado final é mais exato, porém, em contra-partida, esta opção tem uma perda de desempenho.

5.3.2 Juiz Virtual

No sistema Juiz Virtual [JuizVirtual] foi introduzido um método de calibração de câmeras que se enquadra na segunda categoria de métodos segundo Tsai, enumerados na Seção 1.4. O propósito do sistema Juiz Virtual é, dada uma imagem proveniente de uma transmissão de televisão de uma partida de futebol, encontrar a câmera responsável por essa imagem e a partir disto reconstruir a cena 3D, com o objetivo de mover a câmera para outro ponto e redirecioná-la, a fim de poder observar e analisar o mesmo lance a partir de outro ponto de vista.

O método de calibração de câmeras utilizado pelo sistema Juiz Virtual está descrito em [Carvalho+98], detalhando cada passo realizado. Os dados para a calibração da câmera são os pares de pontos (P_i, p_i), onde $p_i = (u_i, v_i)$ são chamados pontos de referência, em coordenadas do sistema da imagem, e $P_i = (x_i, y_i, z_i)$ seus correspondentes no campo real, em coordenadas do sistema do mundo. Os pontos de referência são pontos conhecidos do campo – interseções das linhas do campo e a marca do pênalti, conforme indicado por círculos amarelos na Figura 5.4, que apresenta a mesma imagem da Figura 2.1. Os pontos de objetos da Figura 5.4 (círculos vermelhos) indicam as posições de jogadores, juiz e bola em coordenadas do sistema da imagem e são utilizados para gerar a cena virtual 3D. Todos estes pontos devem ser próprios.

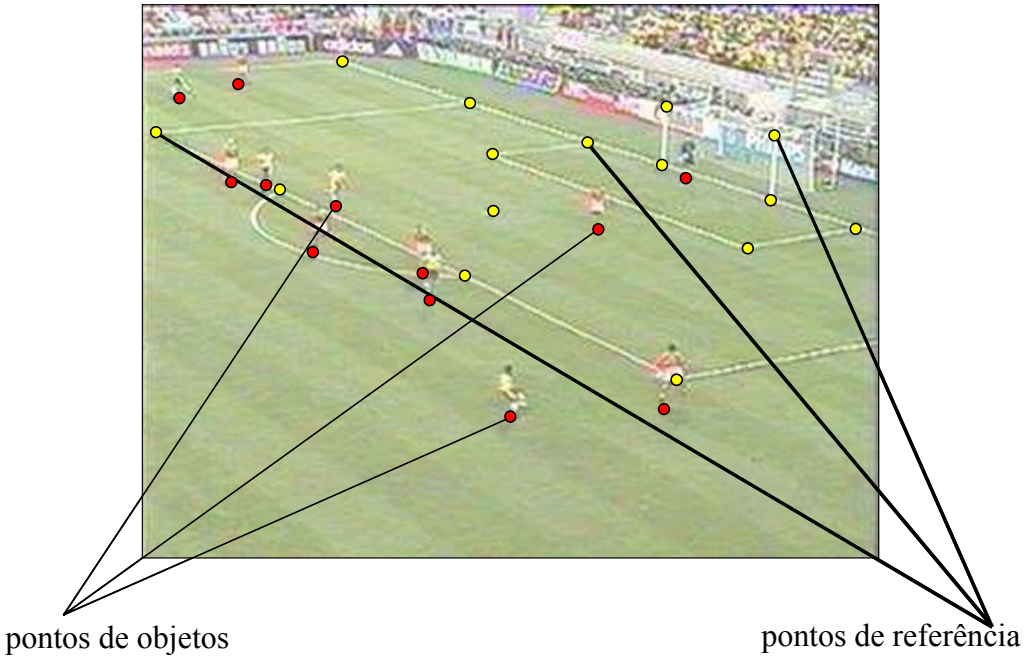


Figura 5.4 – Pontos chave para o sistema Juiz Virtual.

No primeiro passo do algoritmo do Juiz Virtual, é encontrada uma transformação projetiva genérica, sem restrições, que caracteriza uma câmera. O método para encontrar esta transformação é análogo ao descrito na Seção 5.1 para transformações projetivas planares, com a diferença que agora a matriz que representa esta transformação é 3 × 4 denotada por Q, ou seja, é dada por

$$Q\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{q}_1 & q_{14} \\ \boldsymbol{q}_2 & q_{24} \\ \boldsymbol{q}_3 & q_{34} \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} us \\ vs \\ s \end{bmatrix} \tag{5.29}$$

A partir da transformação Q, são encontradas a localização da câmera e a direção ortogonal ao plano onde a imagem é formada. Em seguida, são reajustados os parâmetros da câmera, utilizando os mesmos pontos amostrados usados para encontrar Q, mas certificando-nos de que teremos uma transformação de câmera verdadeiramente geométrica. Primeiro encontramos o centro óptico $O = (x_0, y_0, z_0)$, dado pela solução do sistema

$$\begin{bmatrix} \boldsymbol{q}_1 & q_{14} \\ \boldsymbol{q}_2 & q_{24} \\ \boldsymbol{q}_3 & q_{34} \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.30}$$

Para obter uma estimativa do eixo principal da câmera, primeiro determinamos o ponto *R* do campo, cuja projeção, de acordo com *Q*, é o centro da imagem. Isto é, o ponto *R* tem coordenadas homogêneas (*x*, *y*, *0*, *w*), satisfazendo:

$$\begin{bmatrix} \boldsymbol{q}_1 & q_{14} \\ \boldsymbol{q}_2 & q_{24} \\ \boldsymbol{q}_3 & q_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ w \end{bmatrix} = \begin{bmatrix} u_0 \\ v_0 \\ 1 \end{bmatrix} \tag{5.31}$$

onde u_0 e v_0 são as coordenadas do centro da imagem. O vetor r_3, que resulta da normalização de *R*-*O*, é a estimativa da direção ortogonal ao plano da imagem.

Com estes parâmetros encontrados como estimativas, o problema de calibração fica mais fácil de ser resolvido. Transladam-se os pontos do mundo de *–O*

$$(\tilde{x}, \tilde{y}, \tilde{z}) = (x - x_0, y - y_0, z - z_0) \tag{5.32}$$

e os pontos da imagem de *–C* (ou seja, do simétrico do centro da imagem – (-u_0, -v_0))

$$(\tilde{u}, \tilde{v}) = (u - u_0, v - v_0). \tag{5.33}$$

Portanto, isto faz com que a equação (5.6) possa ser expressa por

$$\begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} f\boldsymbol{r}_1 & 0 \\ f\boldsymbol{r}_2 & 0 \\ \boldsymbol{r}_3 & 0 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ 1 \end{bmatrix}. \tag{5.34}$$

Denotando $q_1 = fr_1$, $q_2 = fr_2$ e $q_3 = r_3$, devido a r_1, r_2 e r_3 serem ortonormais, temos

$$q_1 . q_3 = 0 \tag{5.35}$$

$$q_2 . q_3 = 0 \tag{5.36}$$

$$q_1 . q_2 = 0 \tag{5.37}$$

Seja $q_3 = (a, b, c)$. Se representarmos por *d* e *e* os dois primeiros componentes de $q_1 = (d, e, g)$, a terceira componente *g* satisfaz $ad + be + cg = 0$. Portanto $g = -(a.d + b.e)/c$. Como $q_2 = r_3 \times q_1$, seus componentes são dados por $q_2 = (b.g - e.c, d.c - a.g, a.e - d.b)$. Utilizando estas equações e (5.32), (5.33) e (5.34), além das denotações acima, obtemos as

seguintes expressões para as coordenadas $(\tilde{u},\tilde{v})$ do ponto da imagem correspondente ao ponto (x, y, z):

$$\tilde{u} = \frac{q_1(x-x_0, y-y_0, z-z_0)}{q_3(x-x_0, y-y_0, z-z_0)} = \frac{d.(x-x_0) + e.(y-y_0) - \frac{a.d+b.e}{c} \times (z-z_0)}{a.(x-x_0) + b.(y-y_0) + c.(z-z_0)} \tag{5.38}$$

$$\tilde{v} = \frac{q_2(x-x_0, y-y_0, z-z_0)}{q_3(x-x_0, y-y_0, z-z_0)} = \frac{\left(b.\frac{a.d+b.t}{c} - e.c\right).(x-x_0) + \left(d.c - a.\frac{a.d+b.e}{c}\right).(y-y_0) + (a.e - d.b).(z-z_0)}{a.(x-x_0) + b.(y-y_0) + c.(z-z_0)} \tag{5.39}$$

Ambas expressões acima são lineares em *d* e *e*, que são as únicas variáveis desconhecidas. Estimações para *d* e *e* podem ser obtidas minimizando a soma dos quadrados dos erros cometidos, dados pelas diferenças entre os pontos dados para a calibração deslocados do simétrico do centro e os pontos encontrados pela equação (5.38).

Com o desenvolvimento do sistema e testes realizados com imagens reais, o algoritmo foi aprimorado, sendo realizada uma nova otimização que faz uma busca local nos parâmetros da câmera calculados nas equações (5.30) e (5.31). Isto é feito resolvendo novamente o problema de otimização acima, a fim de minimizar o erro cometido no segundo passo na localização desses parâmetros.

Para o método utilizado neste sistema, os pontos do mundo 3D devem ser não-coplanares e são necessários no mínimo 6 pontos para que possamos calibrar a câmera. Também foi implementado neste sistema o cálculo para encontrar a matriz projetiva planar, utilizado quando o objetivo é realizar medidas no campo, como descrito na Seção 5.1.

5.4 Conclusão

Neste capítulo foram apresentadas duas técnicas simples para obter uma homografia dada por uma transformação projetiva planar a partir de um conjunto de pares de pontos da forma (*P*, *p*), onde *P* é um ponto do mundo da cena e *p* um ponto do plano da imagem. Os pontos *P* devem ser coplanares, assumindo que têm a coordenada *z* igual a zero. O primeiro método permite a utilização de pares onde um dos pontos pode ser impróprio, indicando uma direção ou ponto de fuga.

Em seguida foram descritos dois métodos de calibração de câmeras. No primeiro, o Método de Tsai, foram vistos apenas os passos utilizados pelo algoritmo de acompanhamento de cenas proposto nesta tese, indicando apenas os parâmetros a serem encontrados. Foram

também mostrados os passos para calibrar a câmera através do método desenvolvido no sistema Juiz Virtual, fazendo uma correspondência com o Método de Tsai.

A diferença entre encontrar a homografia e a câmera é que com a câmera podemos reconstruir elementos que estão fora do plano que contém os pontos *P*, como por exemplo as traves do gol. No Juiz Virtual, o posicionamento dos objetos no campo 3D com coordenadas homogêneas (x_i, y_i, 0, w_i) dada sua posição (u_i, v_i) – pontos objetos – na imagem, é calculado pela equação

$$\begin{bmatrix} \boldsymbol{q}_1 & q_{14} \\ \boldsymbol{q}_2 & q_{24} \\ \boldsymbol{q}_3 & q_{34} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 0 \\ w_i \end{bmatrix} = \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \tag{5.40}$$

A cena 3D gerada então para a imagem estática da Figura 5.4 está ilustrada na Figura 5.5. Para calibrar a câmera, o Juiz Virtual usou os pontos de referência mostrados na Figura 5.4, onde as coordenadas dos pontos no campo e na imagem são os listados na Tabela 5.1. Os pontos de objetos estão na Tabela 5.2.

Tabela 5.1 – Pontos de referência da Figura 5.4.

Pontos de Referência				
Campo			**Imagem**	
x	***y***	***z***	***u***	***v***
88,5	13,84	0,00	230	78
88,5	26,69	0,00	134	123
88,5	41,31	0,00	56	158
88,5	54,16	0,00	2	183
94,0	34,00	0,00	147	149
99,5	24,84	0,00	259	133
99,5	43,16	0,00	148	175
105,0	24,84	0,00	307	143
105,0	30,34	0,00	270	155
105,0	37,66	0,00	223	170
105,0	43,16	0,00	192	180
105,0	54,16	0,00	137	198
105,0	30,34	2,44	270	181
105,0	37,66	2,44	225	194

Tabela 5.2 – Pontos de objetos da Figura 5.4.

Pontos de Objetos	
u	***v***
160	61
127	114
83	152
55	158
17	200
227	68
122	126
196	143
234	164
76	133
39	163
44	203

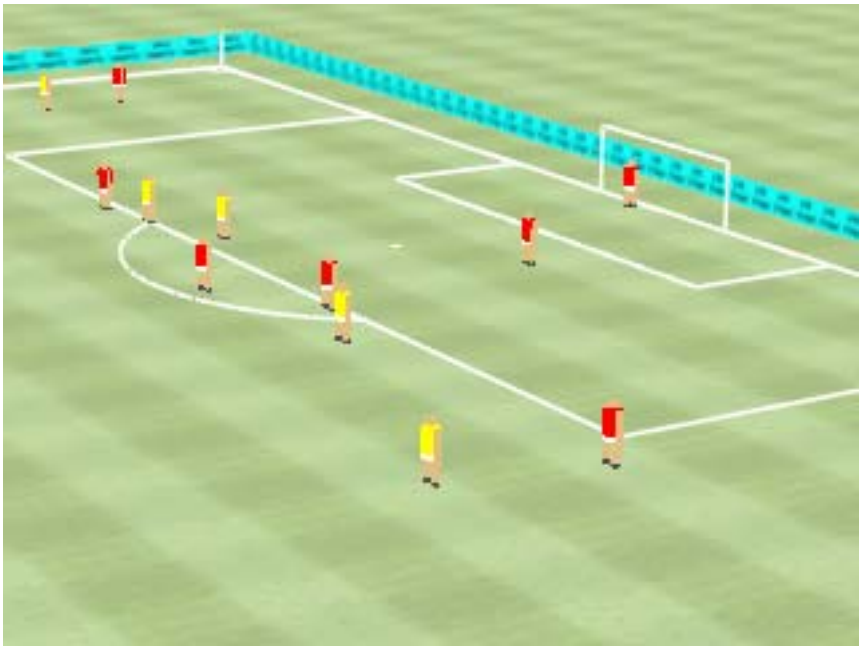


Figura 5.5 – Cena 3D referente à Figura 5.4 gerada pelo Juiz Virtual.

A Figura 5.6 apresenta a visualização da cena de outro ponto de vista.

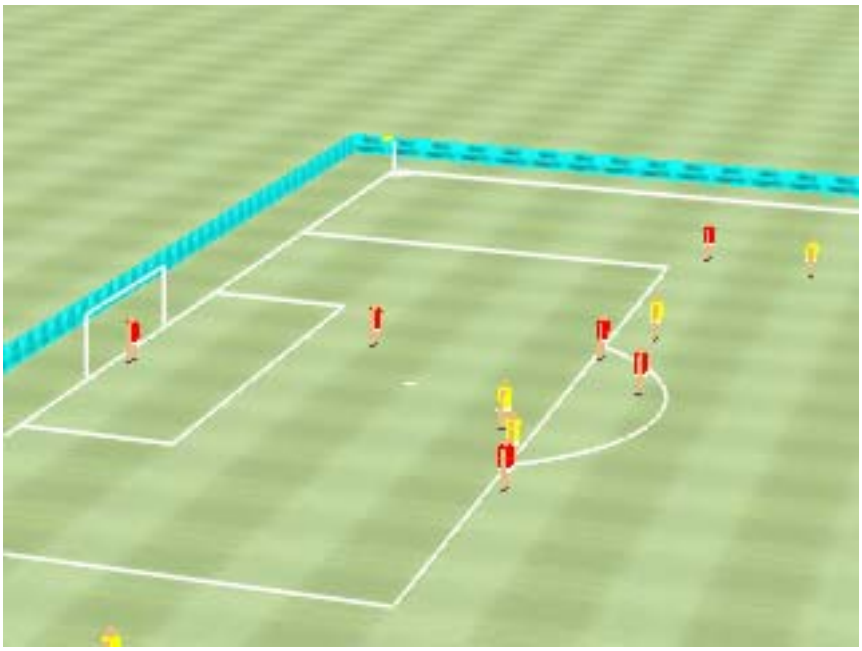


Figura 5.6 – Câmera colocada em outra posição.

A Figura 1.3 ilustra a câmera do sistema Juiz Virtual, onde o telão apresenta a imagem da Figura 5.4, o círculo azul representa a câmera e a sombra indica a parte do campo visível no telão.

Capítulo 6

6 Algoritmo Proposto para Acompanhamento de Cenas

Com os capítulos apresentados até o momento, temos um conjunto de ferramentas suficiente para realizarmos uma calibração automática de câmera. Este capítulo aborda como uma seqüência de cenas pode ser acompanhada através de transformações e filtragens aplicadas às imagens, de detecção e extração de segmentos de retas e do reconhecimento do modelo na imagem. Uma vez extraído o modelo, podemos aplicar as técnicas do capítulo anterior para calibrar a câmera.

O algoritmo proposto aqui é derivado do descrito em [Szenberg+2001], o qual teve seu passo de reajuste aprimorado, incorporou métodos para trabalhar com modelos mais genéricos e adotou pesos nos métodos para encontrar segmentos de retas longos.

Inicialmente, é apresentado um fluxo de passos básicos para calibrar automaticamente a câmera. Em seguida, alguns passos são adicionados a este fluxo, numa tentativa de melhorar o resultado. A seguir, é descrito um conjunto de passos para calibrar a câmera para a segunda imagem de uma seqüência em diante.

A união de todos esses passos resulta no fluxograma apresentado na Figura 1.7, isto é, resulta no algoritmo de calibração automática de câmera através da estimação da nova localização do modelo na imagem por acompanhamento. Convém observar que os passos do algoritmo proposto são independentes, apenas esperam resultados dos passos anteriores. Conseqüentemente, cada passo pode ter seu algoritmo modificado para outro de mesma

finalidade sem ser necessário reescrever todos os outros passos. Isto significa que uma melhoria pode ser feita facilmente, alterando apenas um passo.

6.1 Fluxo de Passos Básico

Nesta seção é apresentado o algoritmo proposto na sua forma básica, composto apenas de passos cujos métodos foram descritos nos capítulos anteriores, sendo discutidos métodos para uni-los seqüencialmente. O algoritmo na forma básica está ilustrado na Figura 6.1, com passos numerados de A1 a A5.

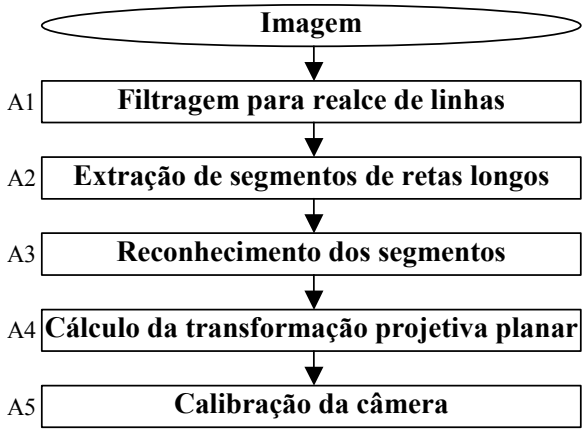


Figura 6.1 – Fluxo de passos básicos para calibração automática de câmera para uma imagem.

Como explicado no Capítulo 2, o passo A1 aplica o filtro *LoG* no negativo da luminância da imagem a fim de realçar linhas e dar suporte a uma segmentação por limiar. O resultado do passo A1 é outra imagem, que apresenta pontos com intensidades maiores como sendo candidatos a estarem sobre algum segmento de reta presente na imagem original, como, por exemplo, a imagem da Figura 2.17. Esta imagem resultante é passada para o passo A2 para que sejam extraídos segmentos de retas longos, candidatos a serem partes da imagem do modelo em questão, conforme explicado no Capítulo 3. O passo A2 tem como saída um conjunto de segmentos de retas, definidos pelos seus pontos extremos, como ilustra a Figura 3.19, que são reconhecidos no passo A3. Isto é, no passo A3 cada segmento de reta extraído é identificado como representante de um dado segmento de reta de um modelo. Este passo está explicado no Capítulo 4 e fornece como resultado um conjunto de relações de representatividade do tipo $\{f_i : F_j$, indicando que tal que o i-ésimo segmento da visualização representa o j-ésimo segmento do modelo $\}$.

A partir das relações de representatividade geradas no passo A3, podemos extrair alguns pontos P do plano que contém o modelo e pontos p do plano da imagem, de modo que p seja a imagem de P. Estes pares de pontos, da forma (P_i, p_i), são extraídos satisfazendo as seguintes condições:

- $P_i = (x, y, 0)$ é a interseção de $F_{x'}$ e $F_{y'}$ (linhas do modelo e o modelo encontram-se no plano $z = 0$);
- $p_i = (u, v)$ é a interseção de f_x e f_y (linhas da visualização do modelo);
- $f_x : F_{x'}$ e $f_y : F_{y'}$.

Além deste conjunto de pares de pontos, podemos ter outros dois pares, correspondentes aos pontos de fuga das direções $\overrightarrow{ox}$ e $\overrightarrow{oy}$ do mundo 3D no plano da imagem. Estes pares são calculados a partir da separação das linhas do modelo em dois conjuntos, um para cada direção, dando origem a dois pontos P_x e P_y, que são impróprios. Encontram-se então os pontos de interseção p_x e p_y (pontos de fuga) dos segmentos de retas dos conjuntos relativos a P_x e P_y, respectivamente, originando os pares (P_x, p_x) e (P_y, p_y). Observe que os pontos p_x e p_y também podem ser impróprios. Todas estas interseções, na verdade, devem ser feitas com as retas suporte dos segmentos de retas, e não apenas com os segmentos.

A Figura 6.2 ilustra os pontos p_i dos pares em relação à Figura 3.18, na qual o resultado obtido do reconhecimento dado o modelo apresentado na Figura 4.10 é

$$f_1 : F_2, \quad f_2 : F_3, \quad f_3 : \varnothing, \quad f_4 : \varnothing, \quad f_5 : F_6, \quad f_6 : F_5, \quad f_7 : F_1.$$

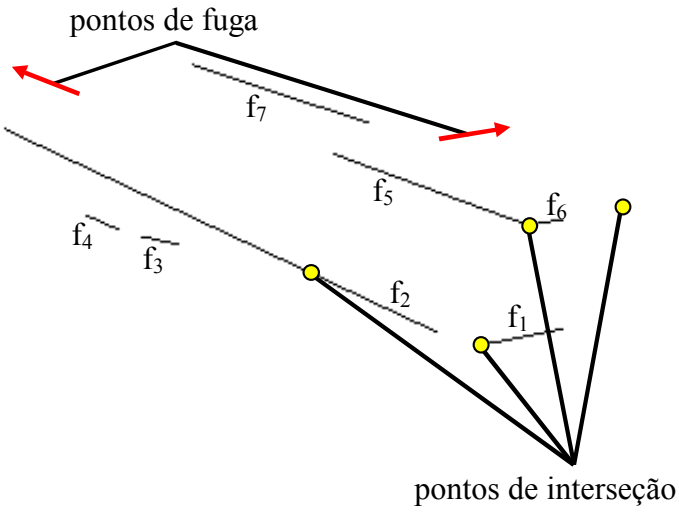


Figura 6.2 – Pontos de interseção e fuga da imagem.

As duas setas da Figura 6.2 indicam os dois pontos de fuga da visualização na imagem, dados pelos dois conjuntos de segmentos – $\{f_2, f_5, f_7\}$ e $\{f_1, f_6\}$. Os círculos indicam os pontos de interseção existentes na imagem.

Com este conjunto de pares de pontos, no passo A4 calculamos uma transformação projetiva planar, usando o método apresentado na Seção 5.1, que permite a utilização de pontos impróprios. Observe que nem todos os segmentos do modelo foram extraídos e apenas os que foram são usados para obter os pares de pontos e conseqüentemente esta transformação. A visualização do modelo segundo esta transformação sobreposta à imagem resultante da filtragem do passo A1 está ilustrada na Figura 6.3.

Figura 6.3 – Projeção do modelo segundo a transformação encontrada no passo A4.

Podemos observar na Figura 6.3 que as projeções dos segmentos através da transformação encontrada sobrepõem bem a imagem do modelo, com exceção do segmento referente ao que não foi extraído, representando a linha F_1 do modelo. Apesar disso, o resultado foi satisfatório, chegando perto do que seria o ideal.

Na calibração de câmera realizada no passo A5, é utilizada como geradora de uma grade de pontos coplanares a transformação encontrada no passo A4. Ou seja, é gerada uma grade de pontos $P = (x, y, 0)$ no plano do mundo que contém o modelo, sendo sua respectiva imagem $p = T(P)$ encontrada através da transformação projetiva planar T. A grade é gerada

no interior da caixa envoltória da parte visível do modelo. O resultado da projeção do modelo do campo de futebol, mais as projeções das traves através da câmera calibrada no passo A5, está ilustrado na Figura 6.4. Observe que, como agora existe uma câmera, foi possível desenhar objetos que encontram-se fora do plano do modelo, que neste caso são as traves do gol.

Figura 6.4 – Projeção do modelo segundo a câmera calibrada no passo A5, com objetos fora do plano do modelo.

Porém, como a sobreposição da projeção do modelo ilustrada na Figura 6.3 apresentou algumas imperfeições, o resultado ilustrado na Figura 6.4 não foi muito satisfatório – as traves não estão muito bem sobrepostas às imagens da trave da cena real. Este resultado pode ser melhorado fazendo um reajuste nessas linhas de acordo com os pontos de maior intensidade que estão próximos às suas projeções na imagem resultante da filtragem do passo A1. Este reajuste é visto na próxima seção, quando aumentarmos o número de passos do algoritmo ilustrado na Figura 6.1.

6.2 Extensão do Algoritmo com Reajuste de Linhas

Nesta seção, o fluxo de passos da Figura 6.1 é aumentado, conforme vemos na Figura 6.5, sendo adicionados os passos B1, B2 e B3, que juntos são responsáveis pelo reajuste das linhas do modelo.

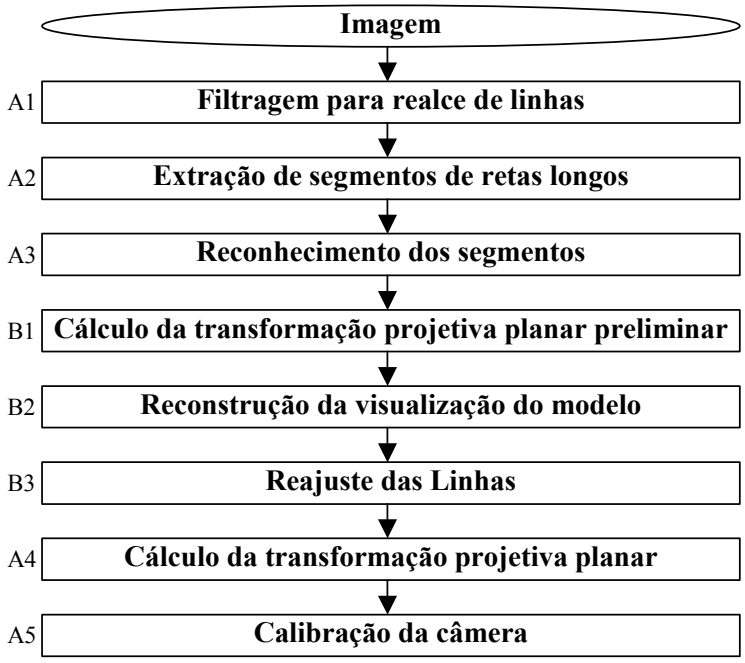


Figura 6.5 – Fluxo de passos estendido, utilizando um reajuste, para calibração automática de câmera para uma imagem.

A diferença entre o fluxo de passos da Figura 6.1 e o da Figura 6.5 é que este tem os passos de cálculo da transformação projetiva planar preliminar, reconstrução da visualização do modelo e reajuste nas linhas da projeção do modelo. A partir desse reajuste, é encontrada uma nova transformação projetiva planar para então calibrar a câmera.

Na verdade, o passo B1 é igual ao passo A4 no fluxo da seção anterior, apenas a transformação é chamada de preliminar pelo motivo já mencionado (somente as linhas extraídas influenciam o cálculo dessa transformação). O passo A4 agora encontra uma transformação a partir do reajuste das linhas reconstruídas, como é visto.

Com a transformação projetiva planar preliminar obtida no passo B1, a imagem do modelo é reconstruída por completo no passo B2, isto é, são estimadas as localizações dos demais segmentos de retas que não foram extraídos no passo A2. O resultado é ilustrado na Figura 6.3.

Os segmentos que foram extraídos no passo A2 são substituídos pelos respectivos segmentos calculados através da transformação projetiva planar preliminar, obtendo os segmentos por inteiro, pois cada segmento extraído pode ser apenas uma parte do segmento

desejado, conforme ilustrado na Figura 3.19. A imagem do modelo é reconstruída simplesmente projetando todo o modelo e ignorando os segmentos extraídos.

O passo B3 é responsável pelo reajuste da projeção das linhas do modelo sobre a imagem. Em princípio, o modelo reconstruído em A3 deveria estar corretamente sobreposto à sua projeção na imagem. Porém, devido a diversos problemas, como presença de objetos, ruídos e erros de precisão, essa sobreposição pode não ocorrer, como foi visto na seção anterior. Para isto, é feito um reajuste nos segmentos do modelo reconstruídos conforme explicado na Seção 3.2.3, utilizando a imagem resultante do passo A1. As faixas de tolerância para os segmentos reconstruídos ilustrados na Figura 6.3 estão mostradas na Figura 6.6 em vermelho, sobrepostas à imagem resultante do passo A1.

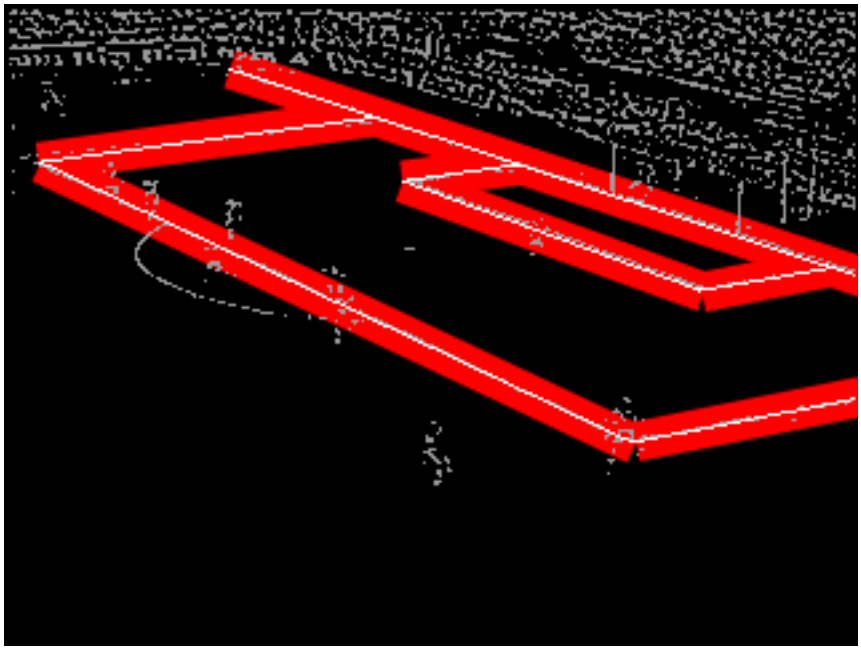


Figura 6.6 – Faixas de tolerância.

A Figura 6.7 ilustra o esquema para reajuste das linhas na imagem com base nas faixas de tolerância. A linha branca é a reconstruída e a linha amarela é localizada a partir dos pontos cinza internos à faixa de tolerância, desenhada na cor vermelho. A linha amarela é localizada através do método de mínimos quadrados (é a reta para a qual a soma dos quadrados das distâncias aos pontos considerados é mínima).

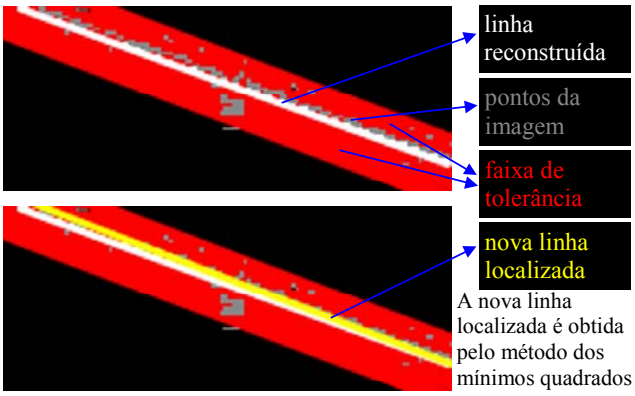


Figura 6.7 – Esquema para reajuste das linhas.

A Figura 6.8 mostra as linhas reconstruídas e reajustadas na imagem.

Figura 6.8 - Linhas reconstruídas e ajustadas sobrepostas à imagem.

Observe que alguns segmentos não são encontrados por inteiro, como a linha lateral da grande área mais ao alto. Isto se deve ao limite aplicado no algoritmo para determinar o segmento de reta a partir da caixa envoltória dos pontos que originaram a reta suporte.

A partir das linhas reajustadas pelo passo B3, o passo A4 encontra uma transformação projetiva planar da mesma forma que no passo B1, com a diferença de agora serem utilizadas todas as linhas do modelo. A Figura 6.9 mostra a projeção das linhas do campo segundo a transformação projetiva planar sobreposta à imagem.

Figura 6.9 – Projeção das linhas segundo a transformação projetiva planar.

A calibração de câmera realizada no passo A5 foi explicada na seção anterior (seu funcionamento é o mesmo nas duas seções). A Figura 6.10 mostra a projeção das linhas do campo com a trave segundo a câmera encontrada pelo Método de Tsai. Observe a diferença entre a Figura 6.4 e a Figura 6.10 – agora a projeção da trave está melhor sobreposta.

Figura 6.10 – Projeção das linhas segundo a câmera encontrada por Tsai.

6.3 Operando com uma Seqüência de Imagens

Na seção anterior, foi apresentado um fluxo de passos para a calibração automática de câmera para apenas uma imagem. Quando temos uma seqüência de imagens, assumindo que não exista nela um corte seco, podemos aplicar métodos de ajuste da câmera para a segunda imagem em diante, levando em consideração coerências entre quadros consecutivos, conforme ilustrado na Figura 6.11.

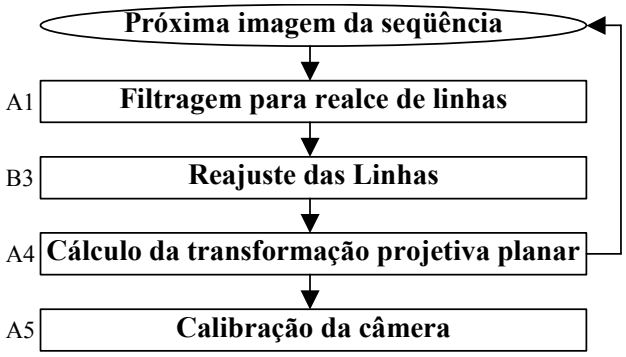


Figura 6.11 – Fluxo de passos para a calibração automática de câmera para a segunda imagem em diante de uma seqüência.

O passo A1 do fluxo de passos da Figura 6.11 é igual ao passo A1 do fluxo de passos da Figura 6.1.

O passo B3, responsável pelo reajuste da projeção das linhas do modelo sobre a imagem, é também utilizado pela demais cenas da seqüência da mesma forma que o passo B3 do fluxo de passos da Figura 6.5. O seu funcionamento é o mesmo para ambos os casos, mas é interpretado de forma diferente. Para a segunda cena em diante, a interpretação do passo de reajuste está ligada à coerência entre quadros. Isto é, as linhas da visualização do modelo de um quadro atual devem estar próximas às linhas correspondentes no quando anterior. Portanto, deve-se apenas fazer um reajuste nessas linhas, utilizando as imagens das linhas do modelo projetado da cena anterior para obter as faixas de tolerância. A Figura 6.12 refere-se a este reajuste aplicado na segunda imagem.

Figura 6.12 - Linhas reconstruídas e ajustadas sobrepostas à segunda cena da seqüência.

A partir das linhas reajustadas pelo passo B3, encontramos uma transformação projetiva planar no passo A4. A Figura 6.13 refere-se à segunda cena da seqüência. A coerência citada pode ser notada pela pequena diferença que existe entre a Figura 6.9 e a Figura 6.13.

Figura 6.13 – Projeção das linhas segundo a transformação projetiva planar (segunda cena).

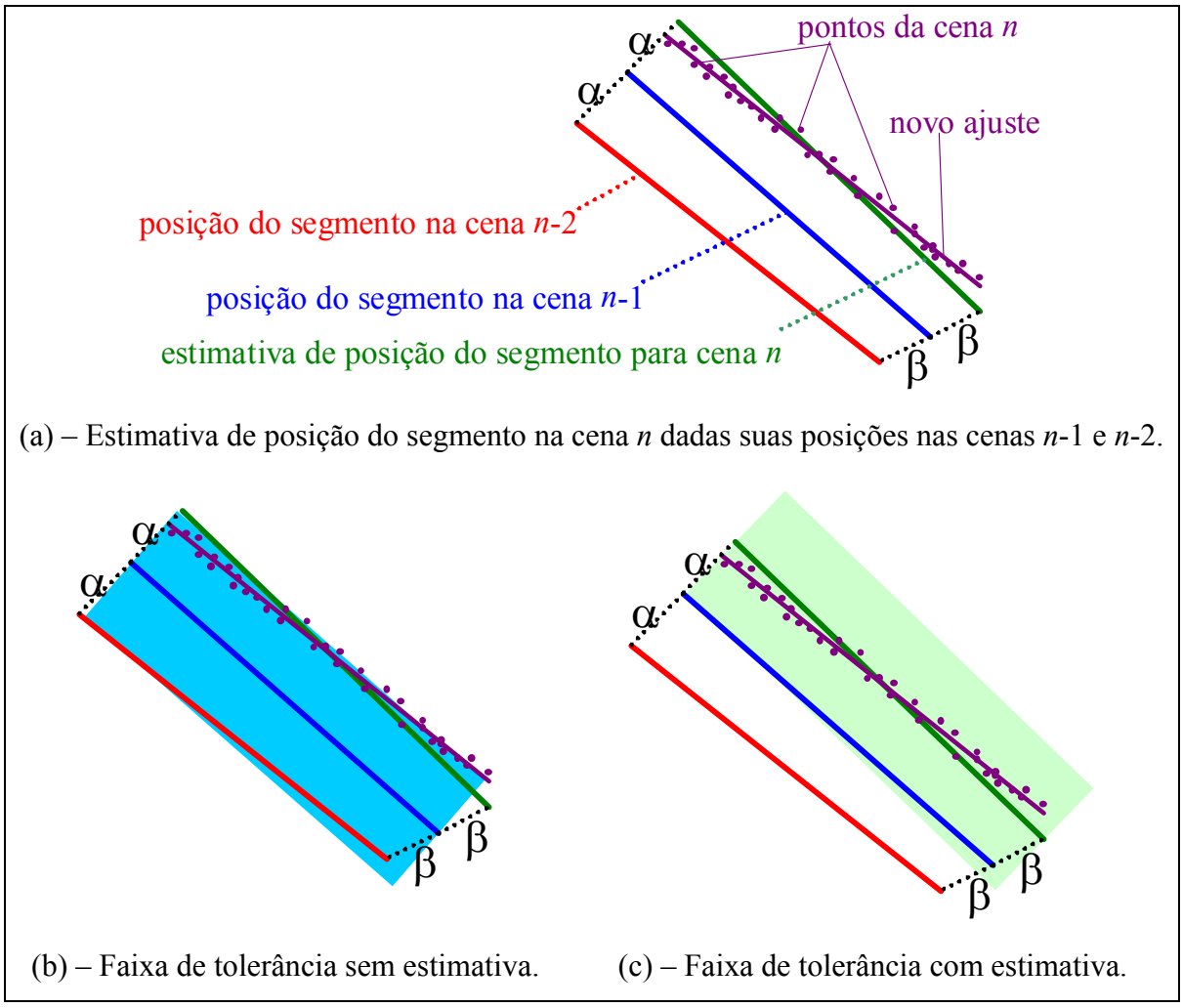
A Figura 6.14 ilustra a projeção do modelo do campo de futebol através da câmera calibrada pelo Método de Tsai no passo A5, da mesmo forma que o passo A5 da Seção 6.1.

Figura 6.14 – Projeção das linhas segundo a câmera encontrada por Tsai (segunda cena).

O reajuste do passo B3 feito para a terceira cena em diante pode ser acrescido de uma estimativa do histórico da localização do modelo na imagem. Podemos observar na Figura 6.15(a) o reajuste desejado de um segmento que na cena anterior (cena n–1) encontrava-se na posição azul e dois quadros atrás (cena n–2) na posição vermelha.

Se utilizarmos o método de reajuste conforme descrito na Seção 3.2.3 sem estimativa alguma, a faixa de tolerância usada na cena atual (cena n) estaria envolvendo o segmento com posição na cena n–1 não enquadrando todos os pontos da cena n (em roxo), como mostra a Figura 6.15(b). Porém, se estimássemos previamente a posição do segmento para a cena n apenas baseando-nos nas duas cenas anteriores, encontraríamos o segmento verde ilustrado na Figura 6.15(a). Esta estimativa baseia-se na tendência de movimento, mantendo a mesma velocidade e direção de deslocamento dos extremos de cada segmento, representados por α e β, através de uma extrapolação. Assim, a faixa de tolerância estaria envolvendo o segmento verde, como mostrado na Figura 6.15(c), enquadrando agora todos os pontos da cena n. Deste modo, portanto, podemos encontrar uma posição melhor para o segmento na cena n, usando a faixa proveniente da estimativa do movimento. Para ficar mais coerente com a tendência de

manipulação da câmera, a extrapolação deve ser feita com os parâmetros da câmera, utilizando quatérnios [Watt+92] para representar os eixos da câmera. Outra opção consiste em fazer uma média ponderada das posições estimadas por estes dois métodos, pois a câmera pode mudar bruscamente a sua trajetória, sem manter coerência com o passado.



(a) – Estimativa de posição do segmento na cena *n* dadas suas posições nas cenas *n*-1 e *n*-2.

(b) – Faixa de tolerância sem estimativa.

(c) – Faixa de tolerância com estimativa.

Figura 6.15 – Reajuste com estimativa de localizações passadas.

Se no passo B3 não for possível reconstruir o modelo, o algoritmo deve ser reiniciado pelo fluxo de passos da Figura 6.1. Um fator que pode ocasionar esta impossibilidade é a existência de cortes secos na seqüência, acarretando falta de coerência entre quadros consecutivos. Neste caso, o coletor usado no reajuste não irá obter pontos dentro das faixas de tolerância dadas a cada visualização da linha do modelo na imagem, fornecendo, assim, uma indicação para a falta de coerência entre os quadros.

O algoritmo final é a união dos fluxos de passos da Figura 6.5 e da Figura 6.11,

resultando na Figura 1.7. Conforme explicado nesta seção, o passo VIII, que equivale ao passo B3, como interpretado para a segunda imagem em diante, garante que apenas uma parte do algoritmo proposto seja usado, otimizando o processo. O passo VIII é feito para a próxima cena da seqüência, na qual são detectados pontos candidatos a estarem sobre algum segmento de reta presente na imagem, da mesma forma que o passo A1 do fluxo de passos da Figura 6.5. Com a imagem resultante do passo VIII e a transformação projetiva planar referente à cena anterior encontrada no passo VII, faz-se um reajuste das linhas do passo VI.

6.4 Heurística para Determinar o Valor de Corte

Um parâmetro importante para o algoritmo é o valor do limiar utilizado no processo de segmentação dos passos I e VIII. Conforme discutido na Seção 2.2.2, métodos para determinação automática deste valor, baseando-se somente em informações da imagem, não foram bem sucedidos. Porém, como o tempo gasto pelo algoritmo para processar uma imagem é muito pequeno, como é visto no capítulo a seguir, é possível utilizar uma heurística baseada na curva que informa quantos segmentos de retas são extraídos para cada valor do limiar de corte na primeira imagem da seqüência. Um exemplo dessa curva está ilustrado na Figura 6.16, referente à imagem da Figura 2.1. Esta heurística procura uma faixa onde a curva apresenta um patamar máximo, e a estimativa é dada pelo ponto máximo dessa faixa. Por exemplo, na Figura 6.16, o valor determinado pela heurística é 20, pois o patamar para valores de limiar entre 15 e 20, inclusive, apresenta valor máximo 7 (o gráfico apresenta um valor máximo porém fora de um patamar, apenas para um determinado valor de limiar).

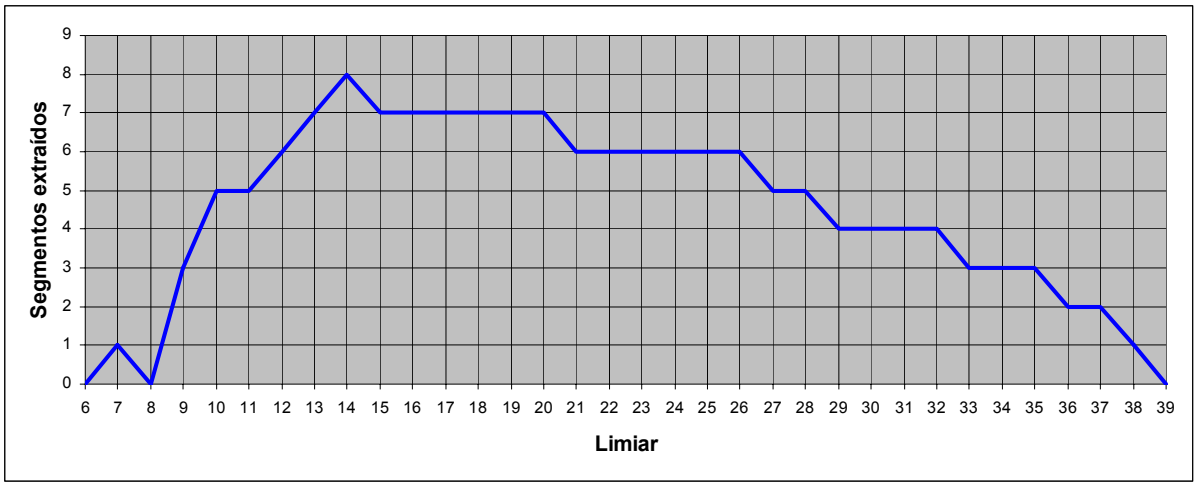


Figura 6.16 – Limiar de corte versus número de segmentos extraídos relativo à Figura 2.1.

Para a Figura 6.17, que apresenta um modelo de um campo de futebol botão, a curva na qual a heurística opera está ilustrada na Figura 6.18. Neste caso, o valor determinado pela heurística é 17.

Figura 6.17 – Imagem de um modelo de campo de futebol de botão.

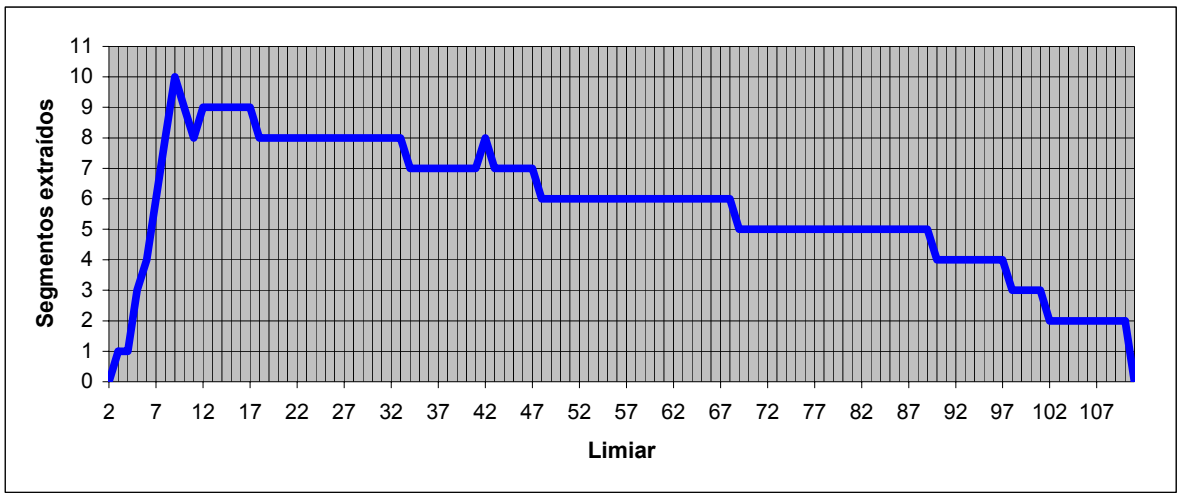


Figura 6.18 – Limiar de corte versus número de segmentos extraídos relativo à Figura 6.17.

Outro exemplo da curva na qual a heurística se baseia está ilustrado na Figura 6.20 relativo à Figura 6.19, que apresenta uma imagem de um modelo diferente de um campo de futebol. Neste caso, o valor encontrado pela heurística é 34.

Figura 6.19 – Imagem de um outro modelo.

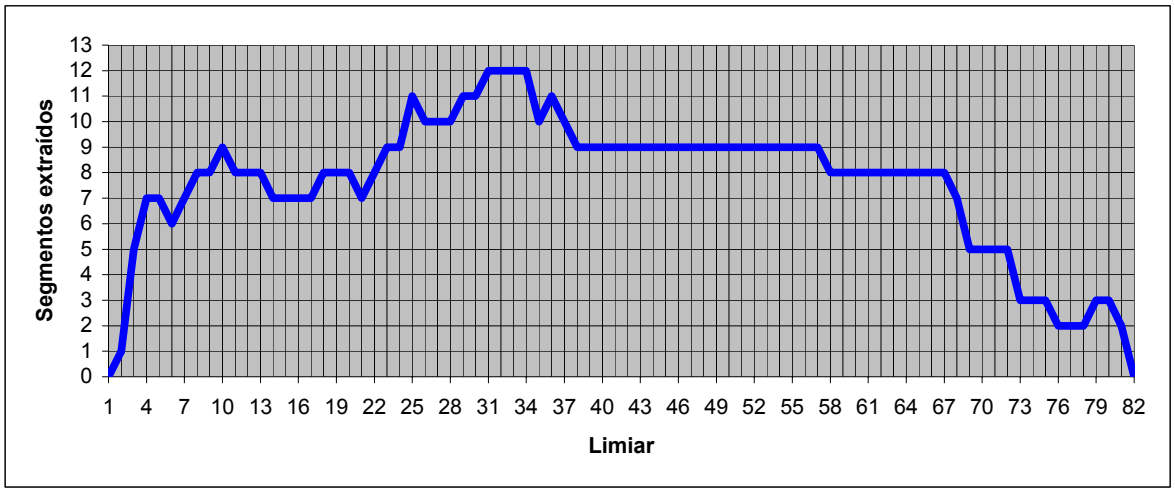


Figura 6.20– Limiar de corte versus número de segmentos extraídos relativo à Figura 6.19.

6.5 Conclusão

Neste capítulo foi descrito o algoritmo proposto nesta tese para acompanhamento de cenas seguido de uma calibração de câmera. Como foi mencionado, quase todos os passos já tinham sido detalhados nos capítulos anteriores, restando apenas esclarecer alguns detalhes, especialmente relativos às conexões dos diversos passos. Além disso, foi mostrada a

necessidade de reajuste para a primeira imagem da seqüência para se obter um resultado melhor.

Os pontos utilizados para encontrar a transformação projetiva planar inicial são obtidos a partir do resultado do reconhecimento do modelo no conjunto de segmentos extraídos da imagem, computando-se interseções entre esses segmentos. Tais interseções podem ocorrer em posições onde encontram-se objetos que as sobrepõem, sendo muitas vezes difícil localizá-las manualmente a partir apenas da imagem. Esta é a vantagem de trabalhar com linhas em vez de pontos, como ocorre no sistema Juiz Virtual com os pontos chamados de pontos de referência (vide Figura 5.4).

Um método proposto para encontrar a transformação projetiva planar permite utilizar pontos de fuga como dados de entrada, o que muitas vezes é necessário, pois podemos ter pontos insuficientes para encontrar tal transformação.

Uma idéia simples foi empregada para reajustar o modelo reconstruído – o uso de faixas de tolerância. Este reajuste serve tanto para a primeira imagem da seqüência, para encontrar posições melhores para os segmentos do modelo reconstruídos pela transformação projetiva planar preliminar, quanto para a segunda imagem em diante, para encontrar a nova localização dos segmentos.

No algoritmo de reajuste utilizado a partir da terceira imagem, foi também apresentada uma técnica de estimativa da nova localização dos segmentos, aumentando o número de pontos usados para encontrar o novo segmento que substituirá o atual. Nos sistemas desenvolvidos nesta tese, esta extrapolação é feita com base na imagem projetada. Embora possivelmente a extrapolação baseada nos parâmetros da câmera seja mais precisa, ela envolve um número grande de parâmetros e cálculos mais complexos, que podem prejudicar o desempenho do método.

Um ponto importante que deve ser enfatizado é a utilização de apenas uma parte do algoritmo para a segunda imagem em diante, aumentando o desempenho. Esta otimização só pôde ser feita pois existe coerência entre quadros – em uma seqüência sem cortes secos, a visualização do modelo em uma imagem deve estar próxima à sua visualização na imagem anterior.

No final, foi apresentada uma heurística para determinar um limiar baseado na curva do número de segmentos extraídos. Essa heurística baseia-se na idéia de que, quanto mais

segmentos de retas o algoritmo retornar, maior a probabilidade de existir um conjunto deles que forme uma imagem do modelo (ainda que seja parcial).

Deve-se observar, finalmente, que em muitas aplicações não há a necessidade de calibrar a câmera, operando-se apenas no plano que contém a imagem. A câmera é necessária apenas para que se possa trabalhar com pontos fora do plano do modelo.

Capítulo 7

7 Implementação e Resultados Obtidos

Visando testar as idéias propostas nesta tese, foram desenvolvidas aplicações para detectar dois modelos: um campo de futebol e um padrão ilustrado na Figura 7.14.

Em todas as aplicações ilustradas, a imagem à esquerda (mais clara) é a original (capturada) e a imagem à direita é o resultado da sobreposição da projeção das linhas do modelo sobre a imagem original utilizando a câmera calibrada pelo algoritmo.

Os testes foram realizados em um computador Pentium III 650 MHz com 512M de memória principal e uma placa de vídeo ASUS V6600 (GeForce 256) Deluxe, que possui embutido um sistema de captura de vídeo, na qual foi ligada uma câmera de vídeo 8mm; e em outro computador, Pentium 4 1.7 GHz com placa de vídeo ASUS V8200 (Geforce 3) Deluxe. Também foi utilizada uma câmera "*webcam*", que é conectada ao computador via USB.

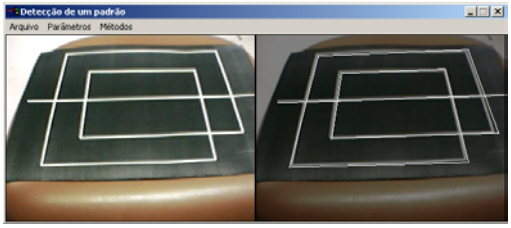
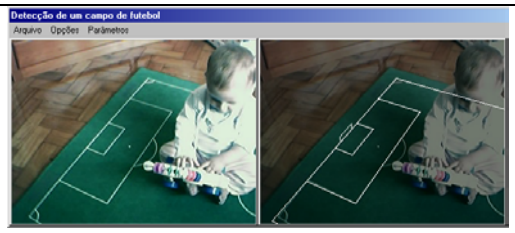
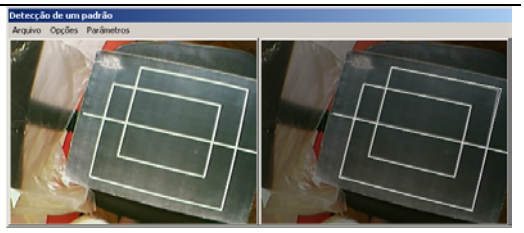
Neste capítulo são apresentados resultados da aplicação do algoritmo, descrevendo primeiro o seu desempenho geométrico e, na subseção seguinte, seu desempenho em tempo de processamento, mostrando os tempos de cada passo do algoritmo.

7.1 Implementações e Testes

Neste trabalho, foram desenvolvidas quatro aplicações que detectam e acompanham modelos em uma seqüência de imagens, sendo duas para modelo de campo de futebol e duas para o modelo dado na Figura 7.14. As duas aplicações para cada modelo diferem na forma

de aquisição das imagens: uma aplicação opera com imagens arquivadas em disco e outra com imagens provenientes diretamente de uma câmera, conforme ilustrado na Tabela 7.1. As aplicações que operam com imagens armazenadas em disco foram desenvolvidas utilizando a biblioteca IUP [IUP], sendo então multiplataforma, enquanto as aplicações que trabalham com captura de imagens foram desenvolvidas utilizando a API do Windows, portanto podendo ser executadas apenas nesta plataforma.

Tabela 7.1 – Tabela de aplicações.

	campo de futebol	outro modelo
imagens em disco		
imagens capturadas		

Aplicações que trabalham capturando diretamente as imagens de uma câmera permitem operar em diversas situações diferentes, como modelo e iluminação, e verificar na prática sua utilização como sistemas de processamento em tempo real, aumentando o número de testes a serem realizados.

Todas as imagens usadas nos testes têm resolução 320 × 240 e são coloridas. A grade usada no passo de eliminação de pontos que não encontram-se sobre um segmento de reta (passo II do algoritmo) divide as imagens de forma que as células tenham altura e largura iguais a 10 e o fator que determina se uma célula tem disposição de pontos segundo um segmento de reta igual a 20. O valor de corte usado na segmentação é determinado pela heurística apresentada na Seção 6.4. Em relação à condição de paralelismo, os valores são $\eta = 2$ e $\rho = 0{,}81$, conforme descrito na Seção 4.2.2. Finalmente, a largura das faixas de tolerância têm espessura igual a 8. Todos estes parâmetros, com exceção de η e ρ, podem ser alterados pelos programas de testes.

Como modelos do campo de futebol, foram empregados campos verdadeiros, através

de imagens de jogos de futebol transmitidos por emissoras de televisão, e uma metade de um tabuleiro de futebol de botão. Esses dois modelos de campo, além de naturalmente apresentarem medidas diferentes, têm proporções desiguais. O outro modelo, diferente do campo de futebol, não apresenta simetria.

7.2 Desempenho Geométrico

Nesta seção são apresentados resultados referentes ao desempenho geométrico do algoritmo. São descritos resultados para modelos de um campo de futebol e para o modelo da Figura 7.14. Foram testadas as quatro aplicações citadas no início deste capítulo.

7.2.1 Detecção de um Campo de Futebol

Para o caso da detecção de um campo de futebol, duas situações foram testadas: uma em que as imagens eram de uma partida real de futebol e outra em que foi filmado um tabuleiro de futebol de botão, utilizando uma câmera de vídeo 8mm e uma "*webcam*".

- Imagens de uma partida real de futebol

Nestas imagens estão contidos jogadores, bola e textura de gramado, além de estarem presentes ruídos devido à transmissão do sinal de vídeo e distorções devido às câmeras. Na aplicação utilizando este tipo de dados, todas as figuras são lidas do disco rígido e armazenadas em memória principal na forma de um vetor de imagens.

Um exemplo do funcionamento do algoritmo para este tipo de imagem é dado na Figura 7.1. Esta imagem é a primeira de uma seqüência composta de 67 imagens. Vale lembrar que, sendo ela a primeira imagem de uma seqüência, todos os passos do algoritmo proposto foram realizados – filtragem, detecção e extração de segmentos de retas longos, reconhecimento de padrões, transformação projetiva planar preliminar, reconstrução do modelo, reajuste das linhas, transformação projetiva planar e calibração de câmera.

Nas Figuras 7.2, 7.3 e 7.4, vemos os resultados obtidos pelo algoritmo utilizando apenas os passos de filtragem, reajuste das linhas, transformação projetiva planar e calibração de câmera. Estas figuras são relativas às cenas de número 23, 45 e 67, respectivamente. Podemos observar, em todas as figuras, resultados visualmente bons.

A Figura 7.5 apresenta o resultado da execução de todos os passos do algoritmo para a última imagem de uma seqüência como se fosse a primeira. A diferença entre os resultados

apresentados nas Figuras 7.4 e 7.5 está ilustrada na Figura 7.6.

Figura 7.1 – Detecção de um campo real (primeira imagem de uma seqüência).

Figura 7.2 – Detecção de um campo real (imagem nº 23 de uma seqüência).

Figura 7.3 – Detecção de um campo real (imagem nº 45 de uma seqüência).

Figura 7.4 – Detecção de um campo real (imagem nº 67 – última – de uma seqüência).

Figura 7.5 – Resultado para a última imagem da seqüência, utilizando todos os passos do algoritmo.

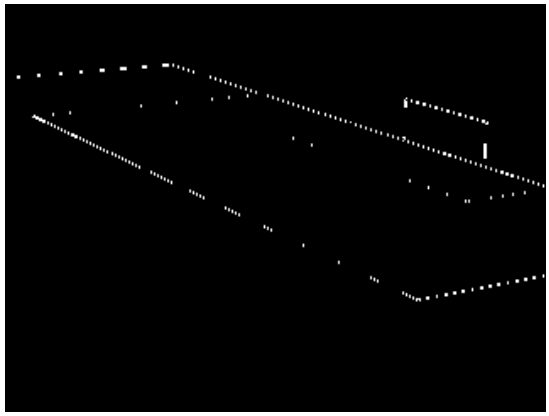


Figura 7.6 – Diferença dos resultados obtidos ilustrados nas Figuras 7.4 e 7.5.

Observe que a diferença encontrada na Figura 7.6 é sutil, mas existe. Ela se deve ao fato de que o reajuste das linhas realizado na obtenção do resultado apresentado na Figura 7.4 baseia-se em um conjunto de linhas obtido na cena anterior da seqüência, enquanto o da Figura 7.5 baseia-se na extração da própria figura de origem. Isto pode acarretar uma diferença nas faixas de tolerância no momento de encontrar pontos para o reajuste das linhas.

Para verificar a precisão do algoritmo, foi utilizada uma seqüência de imagens artificiais, exemplificada nas Figuras 7.7 e 7.8, referentes à primeira e à última (27ª) imagem, respectivamente.

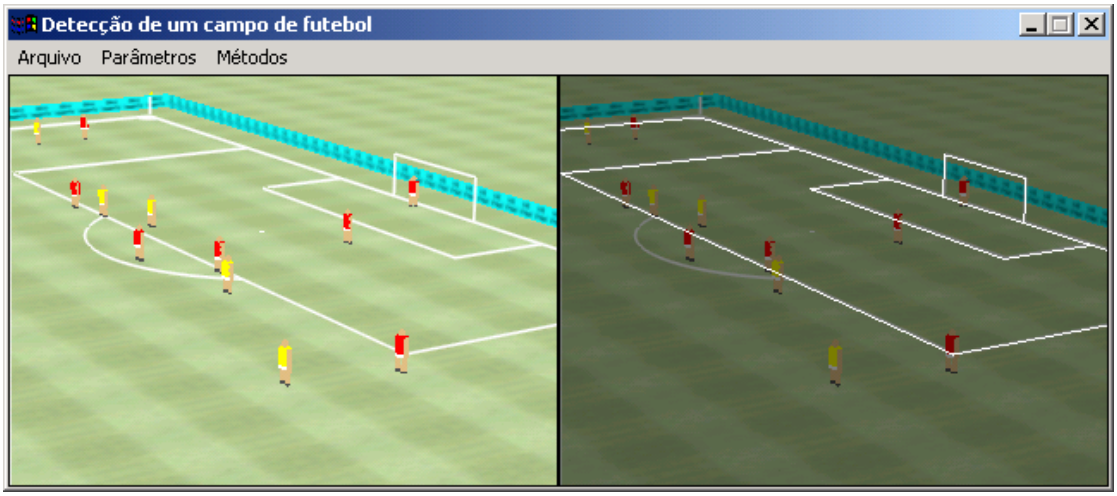


Figura 7.7 – Primeira imagem de uma seqüência artificial.

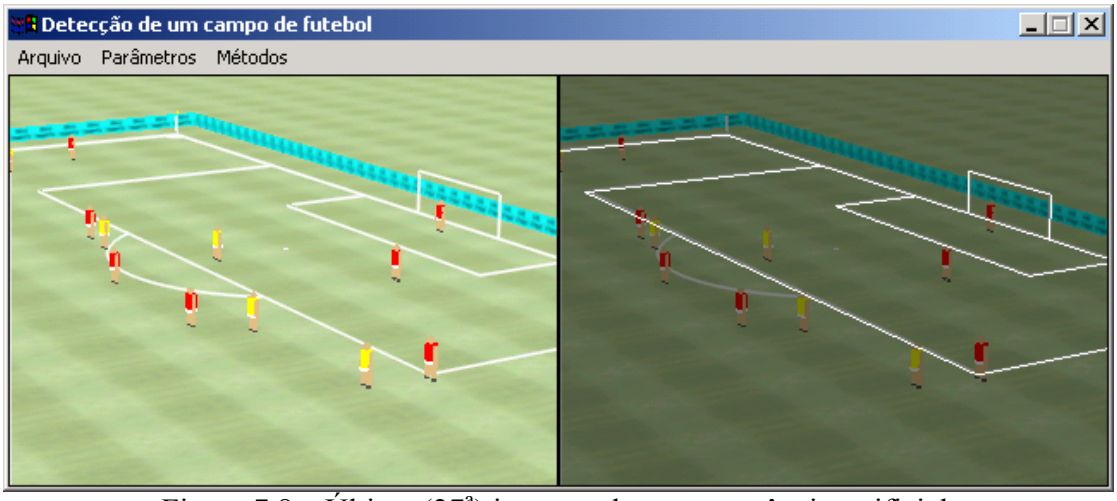


Figura 7.8 – Última (27ª) imagem de uma seqüência artificial.

Na seqüência artificial de imagens, os erros visuais não podem ser notados. Na seqüência real, existem pequenos erros na sobreposição das linhas reconstruídas (em branco) sobre as linhas das imagens. Os erros numéricos sobre a seqüência artificial estão mostrados

na Tabelas 7.2 e 7.3, referentes à primeira e à última imagem, respectivamente. Estas tabelas apresentam as coordenadas projetadas corretas e os resultados a partir do algoritmo proposto nesta tese (coordenadas reconstruídas). As comparações mostram que o erro, para cada ponto, nunca é maior que 2 pontos, com erro médio por volta de 0,5 ponto. Estes pequenos erros resultam principalmente da natureza discreta das imagens em baixa resolução (320 × 240).

Tabela 7.2 – Comparação entre as coordenadas corretas e reconstruídas (primeira imagem).

Pontos do Campo			**Coordenadas Corretas**		**Coordenadas Reconstruídas**		**Erro** (distância euclidiana)
x	***y***	***z***	***u***	***v***	***u***	***v***	
105,0	68,00	0,00	81,707	216,584	81,731	215,972	0,612
88,5	13,84	0,00	230,117	78,133	228,747	77,525	1,499
88,5	54,16	0,00	1,236	183,463	0,424	183,197	0,854
99,5	24,84	0,00	259,039	134,206	258,566	133,815	0,614
99,5	43,16	0,00	146,690	174,826	146,067	174,484	0,711
105,0	30,34	0,00	269,817	155,102	269,629	154,697	0,446
105,0	30,34	2,44	270,921	181,066	270,215	180,863	0,735
105,0	37,66	2,44	224,101	194,645	223,291	194,407	0,845
105,0	37,66	0,00	223,405	170,271	223,082	169,876	0,510
						Erro Médio	0,696

Tabela 7.3 - Comparação entre as coordenadas corretas e reconstruídas (última imagem).

Pontos do Campo			**Coordenadas Corretas**		**Coordenadas Reconstruídas**		**Erro** (distância euclidiana)
x	***y***	***z***	***u***	***v***	***u***	***v***	
105,0	68,00	0,00	97,167	205,940	96,791	205,585	0,517
88,5	13,84	0,00	243,883	66,434	243,549	66,022	0,530
88,5	54,16	0,00	16,101	173,174	15,655	172,623	0,709
99,5	24,84	0,00	273,344	124,029	273,125	123,715	0,382
99,5	43,16	0,00	160,672	164,798	160,366	164,421	0,486
105,0	30,34	0,00	284,160	145,173	283,992	144,914	0,309
105,0	30,34	2,44	285,241	171,290	284,886	171,090	0,407
105,0	37,66	2,44	238,127	184,768	237,744	184,538	0,447
105,0	37,66	0,00	237,462	160,349	237,252	160,063	0,355
						Erro Médio	0,452

- Câmera tipo "*webcam*"

Este tipo de câmera apresenta alto fator de distorção de lente. Foi utilizado aqui um tabuleiro de futebol de botão e as razões de medidas são totalmente diferentes de um campo real, como usado no item anterior. Apesar do algoritmo proposto nesta tese não tratar

distorção de lente, ele conseguiu detectar muito bem as linhas do campo, como podemos constatar na Figura 7.9. Nesta aplicação, as imagens processadas são provenientes diretamente da câmera, sem armazenamento no disco rígido.

Figura 7.9 – Exemplo de detecção de um campo de futebol.

Resultados onde a figura contém objetos diferentes da linha do campo ou parte do campo sofre oclusão estão ilustrados nas Figuras 7.10 e 7.11. Na Figura 7.10 existe um bebê (minha filha) segurando um objeto (brinquedo). Na Figura 7.11, o bebê está escondendo parte das linhas do campo. Nestas duas figuras, os resultados foram bons pois as linhas do campo desenhadas usando a câmera calibrada no algoritmo sobrepõem quase perfeitamente as linhas do campo do tabuleiro de futebol de botão. Observe que estão desenhadas também as traves, aparentemente bem localizadas.

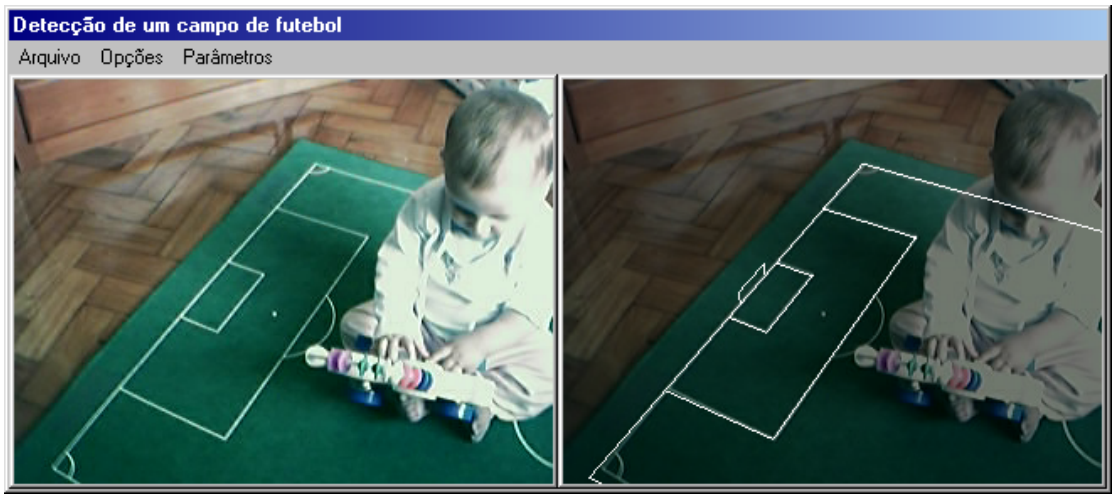


Figura 7.10 – Exemplo de detecção de um campo de futebol com interferência.

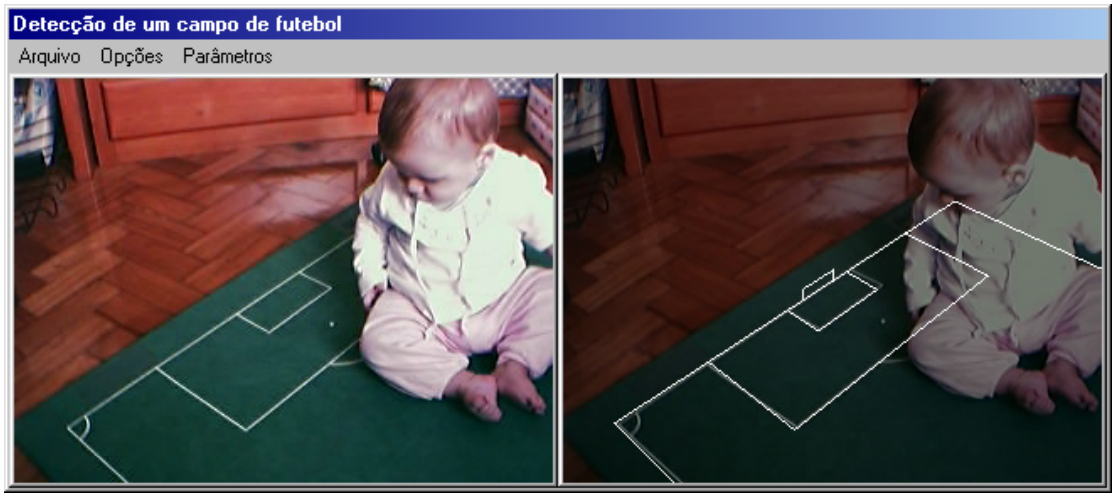


Figura 7.11 – Exemplo 2 de detecção de um campo de futebol com interferência.

- Câmera 8mm

A Figura 7.12 mostra o protótipo usado para testar o algoritmo utilizando uma câmera de vídeo 8mm colocada sobre um tripé direcionada para um tabuleiro de futebol de botão cortado ao meio. O resultado do algoritmo está na Figura 7.13.

Figura 7.12 – Protótipo para testar o algoritmo usando uma câmera 8mm.

A imagem da esquerda da Figura 7.13 é exatamente o que a câmera que aparece na Figura 7.12 capturou e a imagem da direita é o campo gerado pela câmera calibrada pelo algoritmo sobreposto ao tabuleiro de futebol de botão.

Figura 7.13 – Resultado do algoritmo usando a câmera posicionada conforme a imagem da Figura 7.12.

Os resultados utilizando este tipo de câmera são bem semelhantes aos obtidos quando usamos a "*webcam*", exceto por ela não possuir uma distorção de lente tão grande quanto a "*webcam*" e o tempo de captura ser bem menor (devido ao *hardware*), suavizando o resultado da sobreposição do campo às figuras quando feita uma animação.

7.2.2 Detecção de Outro Modelo

Um outro modelo ou padrão utilizado nas aplicações para testar o algoritmo proposto está ilustrado na Figura 7.14. O objetivo ao testar com este padrão é verificar o comportamento do algoritmo quando o padrão não apresenta simetria e possui linhas se intersectando, como por exemplo, linhas F_5 e F_8.

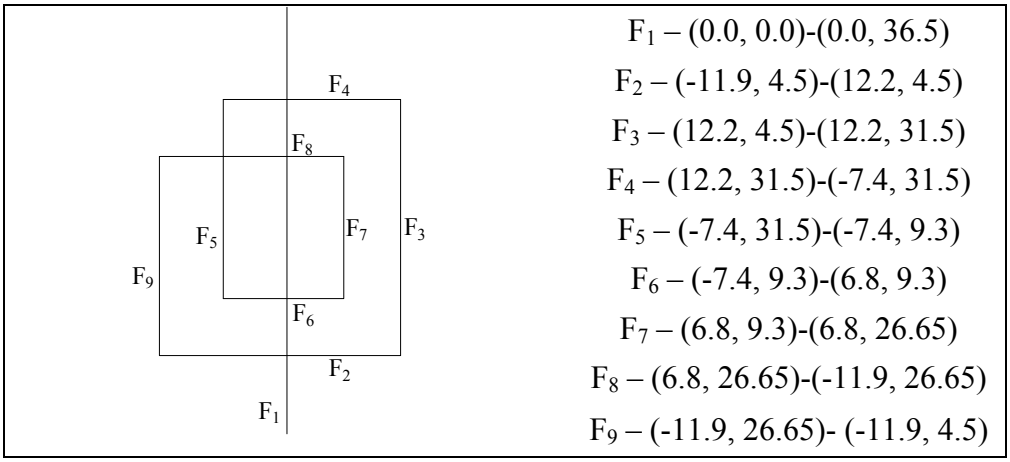


Figura 7.14 – Exemplo de outro modelo ou padrão.

A Figura 7.16 apresenta o resultado da extração de segmentos de retas longos da imagem com base na Figura 7.15, onde vemos a imagem filtrada (negativo e *LoG*) e segmentada (valor de corte igual a 41). Podemos observar que alguns segmentos estão "incompletos". Isto se deve à divisão da imagem em retângulos, em alguns dos quais a condição da razão entre o maior e o menor autovalor não foi satisfeita.

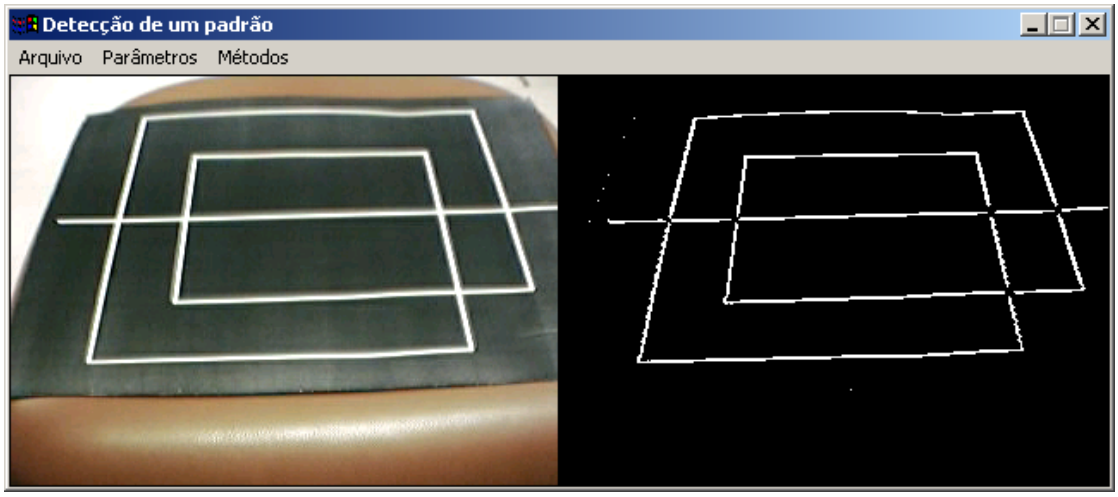


Figura 7.15 – Resultado da filtragem e segmentação com valor de corte igual a 41.

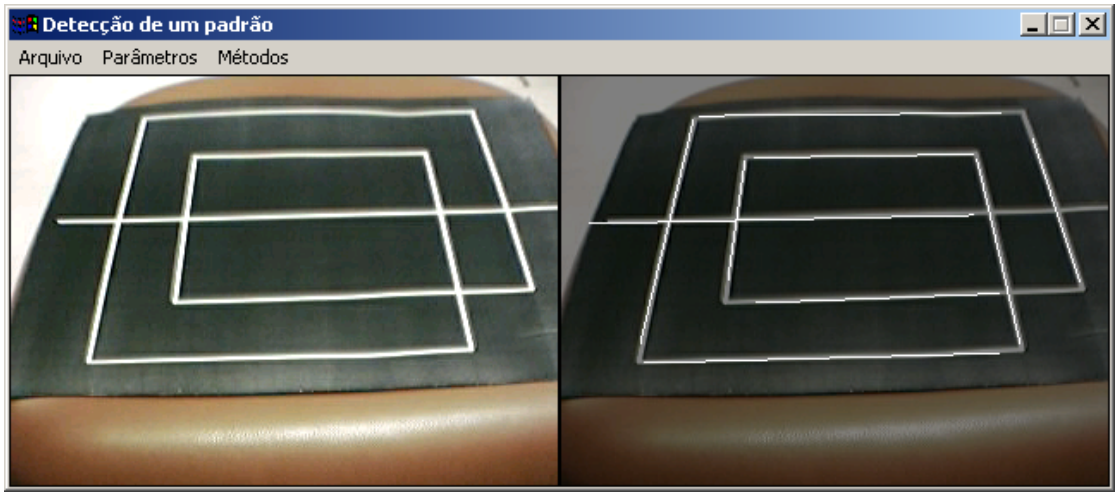


Figura 7.16 – Linhas extraídas.

Observe que a linha do modelo visualizado localizada mais acima na figura não é um segmento de reta. Isto ocorre porque a imagem original é a visualização de uma impressão do modelo em um papel que foi colocado em cima de uma cadeira sem estar esticado. Assim, devido às suas dobras, ocorreram deformações no modelo impresso. Além disso, existe ainda a distorção da lente da câmera ("*webcam*").

Tendo sido extraídos os segmentos de retas longos, foi encontrada a transformação projetiva planar preliminar e então reconstruído o modelo: o resultado pode ser visto na Figura 7.17. Observe que o segmento de reta do modelo reconstruído mais acima na imagem está representando bem a linha correspondente na visualização, que apresenta uma certa curvatura e dobra.

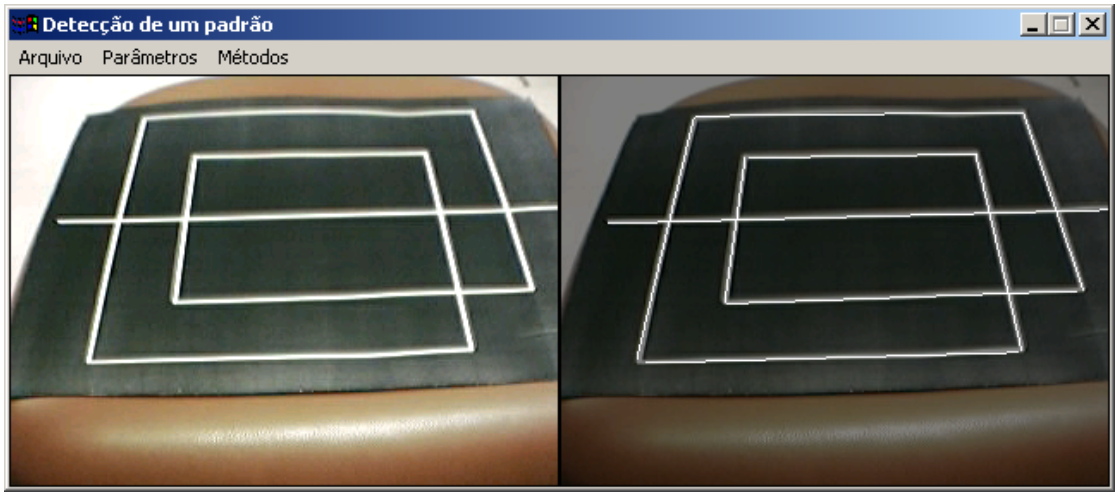


Figura 7.17 – Modelo reconstruído.

A Figura 7.18 ilustra a transformação projetiva planar encontrada a partir do modelo reconstruído. Podemos observar que existe uma diferença na sobreposição do modelo projetado pela transformação e a visualização, mais especificamente na parte inferior esquerda da imagem. Isto ocorre devido às dobras sofridas pelo papel e à distorção da câmera. Esta região é a mais próxima da câmera, sofrendo portanto maiores distorções.

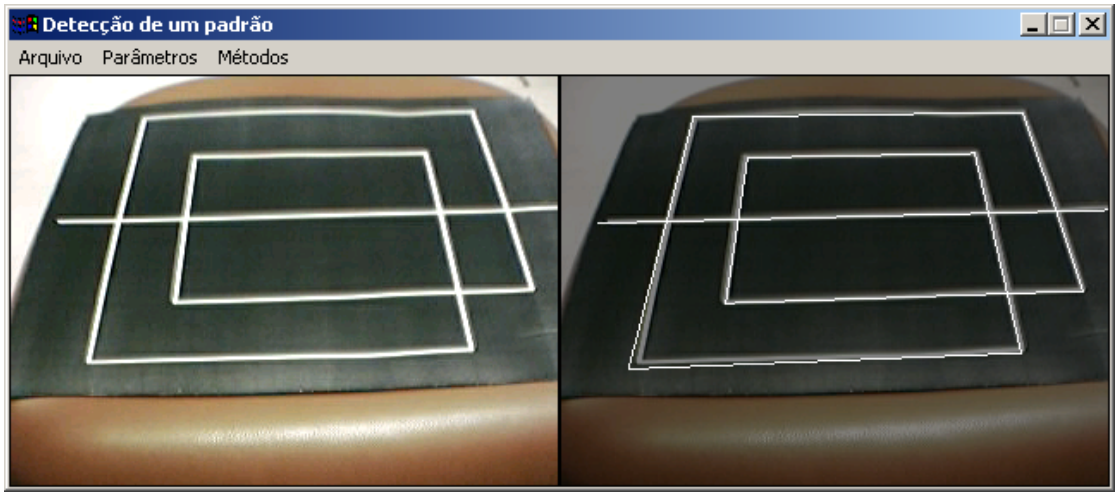


Figura 7.18 – Transformação planar projetiva.

Finalmente, a Figura 7.19 mostra o resultado obtido pela visualização do modelo utilizando a câmera calibrada pelo método de Tsai. O erro cometido sobre a transformação projetiva planar, discutido acima, foi distribuído em todos os pontos.

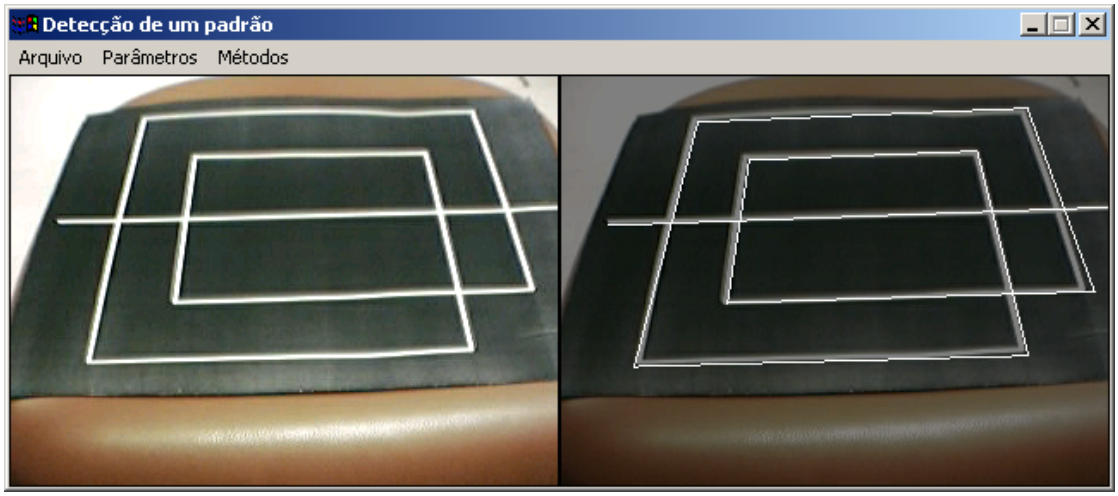


Figura 7.19 – Câmera calibrada pelo método de Tsai.

A Figura 7.20 apresenta outro resultado da detecção desse padrão. Neste caso, o papel foi colocado com cuidado para não haver dobras e com uma visão quase perpendicular ao plano do modelo. A sobreposição do modelo pela câmera calibrada pelo método de Tsai neste caso foi bem feita, exceto em uma região mais acima e à direita. Podemos observar que nesta região encontra-se o ponto de interseção de dois segmentos de retas do modelo que apresentam uma leve curvatura devido à distorção da lente da câmera.

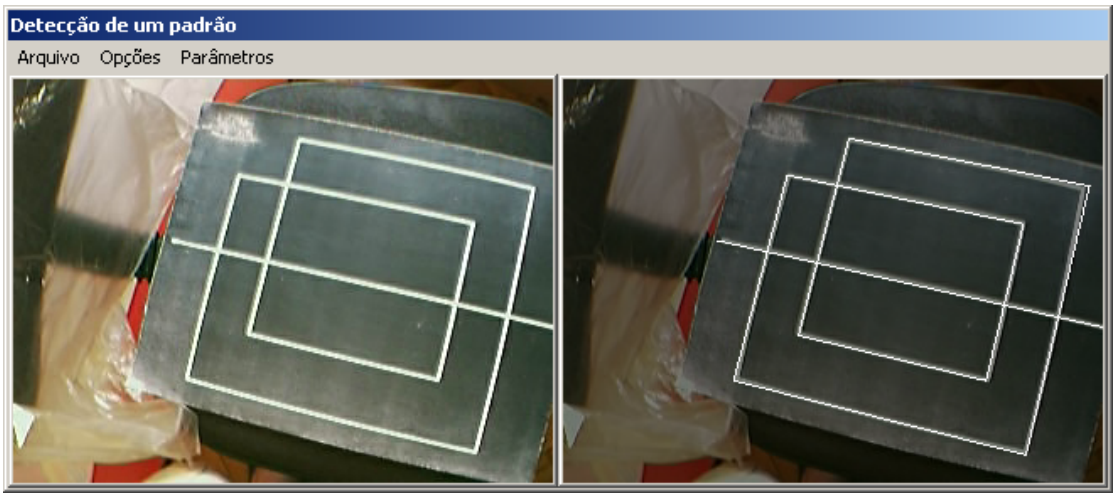


Figura 7.20 – Outra detecção do padrão diferente do campo de futebol.

7.3 Desempenho de Tempo de Processamento

Foram tomados os tempos gastos para processar uma seqüência de 67 imagens com resolução 320 × 240, armazenadas na memória principal, relativo ao primeiro exemplo da Seção 7.2.1, desenhando, ou não, o modelo sobreposto a cada imagem através da câmera encontrada. Foram usadas apenas 67 imagens para não ocorrer paginação na memória, evitando-se, assim, contaminar a avaliação de eficiência do algoritmo com tempos de acesso a disco. Esses tempos estão mostrados na Tabela 7.4. Observe que estes tempos estão bem abaixo do necessário para um processamento em tempo real, que seria de 2,23 segundos (30 *fps*).

Tabela 7.4 – Tempos de processamento de uma seqüência de 67 imagens.

	Pentium III 650 MHz	Pentium 4 1.7GHz
Não desenhando o modelo	0,59 segundos (113,6 *fps*)	0,19 segundos (352,6 *fps*)
Desenhando o modelo	1,0 segundos (67 *fps*)	0,5 segundos (134 *fps*)

Como apenas para a primeira imagem da seqüência são executados todos os passos de do algoritmo (com exceção do passo VIII), a maior parte do tempo acima é gasta no processamento dos passos VI, VII, VIII e IX, que é feita para cada imagem a seguir. O gráfico da Figura 7.21 mostra a contribuição percentual média de cada passo para o tempo total, obtida processando a seqüência 40 vezes. Observamos que a maior contribuição é proveniente do passo VIII, que executa o filtro *LoG* sobre a imagem inteira. Uma possibilidade de melhoria seria aplicar o filtro seletivamente, apenas em regiões próximas às linhas encontradas na imagem anterior.

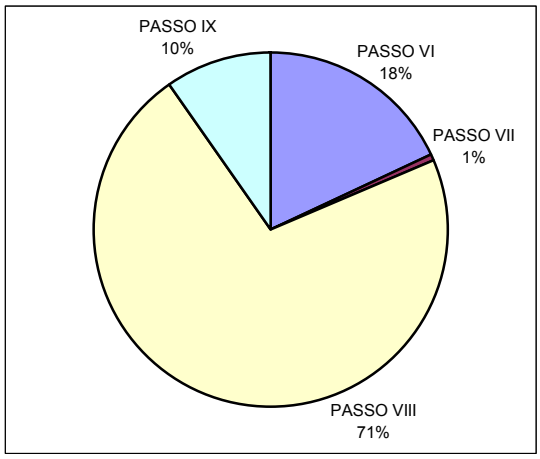


Figura 7.21 – Comparação de tempos entre os passos VI, VII e VIII e IX.

7.4 Problemas Ocorridos

Como visto nos casos acima, embora o algoritmo tenha conseguido acompanhar bem os modelos nas cenas, existem situações em que ele falha. Entre estas situações, pode-se listar:

- A primeira imagem da seqüência a ser processada deve possuir um conjunto de segmentos de retas, possíveis de serem extraídos, capaz de possibilitar o reconhecimento do modelo. No caso onde o modelo é um campo de futebol, no mínimo quatro linhas do campo devem ser extraídas da imagem, sendo uma, linha de fundo (linha que contem as traves do gol). Para a segunda imagem em diante, o reajuste deve ser realizado com sucesso (possuir pontos capaz de localizar o novo segmento) em 4 linhas, duas paralelas a cada direção $\overrightarrow{ox}$ e $\overrightarrow{oy}$;
- Linhas não nítidas: em algumas imagens, as linhas do modelo podem não estar muito nítidas, o que dificulta suas extrações, como é o caso na Figura 7.22;

Figura 7.22 – Linhas do campo sem nitidez.

- Variação na iluminação: quando uma parte da cena onde encontra-se o modelo está iluminada e outra com sombra, a determinação do valor de corte no processo de segmentação é dificultada. Este é o caso da Figura 7.23.

Figura 7.23 – Parte da cena com iluminação e outra com sombra.

7.5 Conclusão

Neste capítulo foram apresentados testes realizados com as aplicações desenvolvidas, onde estão implementados os métodos apresentados em cada capítulo e o algoritmo proposto nesta tese. Foram feitos testes para verificar seu desempenho geométrico, analisando os erros visuais cometidos em diversos passos sobrepondo os resultados às imagens originais, além de serem analisados os desempenhos em tempo de processamento para uma seqüência de imagens, mostrando quanto tempo foi gasto para cada passo do algoritmo.

A partir dos testes de desempenho geométrico, podemos concluir que os resultados foram bem satisfatórios, com alguns casos em que os erros não são perceptíveis visualmente. De acordo com os valores apresentados nas Tabelas 7.2 e 7.3, este fato ficou comprovado, pois os erros médios são menores que 0,7 ponto e o erro máximo foi 1,5 ponto.

O algoritmo também apresentou bons resultados mesmo quando parte da imagem do modelo está oclusa e existem pessoas e objetos na cena. Isto foi comprovado nas imagens provenientes de uma transmissão de partida de futebol, na qual existem jogadores se movendo sobre o gramado, e pelos resultados apresentados nas Figuras 7.10 e 7.11, onde há uma pessoa cobrindo grande parte do modelo.

Mesmo o modelo do campo de futebol sendo coplanar, as linhas correspondentes às traves do gol, que localizam-se fora do plano das linhas do campo, foram bem sobrepostas às traves presentes na imagem original, comprovando o desempenho geométrico da câmera calibrada. As traves, assim como qualquer outro objeto que esteja fora do plano do modelo, só podem ser projetadas porque a câmera foi calibrada no passo IX do algoritmo. Com a transformação projetiva encontrada no passo VII, é possível somente desenhar objetos que se encontrem no mesmo plano do modelo.

Os testes realizados com outro modelo, não simétrico, também foram bem satisfatórios, levando em consideração que este modelo foi impresso em um papel que não está totalmente planar.

Os tempos mostrados na Tabela 7.4 e a taxa de processamento médio em quadros por segundo (*fps*) provam que o algoritmo é capaz de operar em tempo real, pois estão bem acima da taxa de 30 *fps*. Na Figura 7.21, podemos observar que o algoritmo leva mais tempo no passo VIII, onde é feita a filtragem através de uma convolução e uma segmentação. Este passo é o mais lento pois é preciso percorrer toda a imagem para fazer a convolução. O segundo passo que toma mais tempo é o VI, responsável pelo reajuste das linhas. Isto ocorre pois a imagem também deve ser inteiramente percorrida, encontrando-se, para cada ponto, a linha do modelo reconstruído na cena anterior que esteja mais próxima dele.

Uma observação importante é que, apesar de não levar em consideração distorções ópticas da câmera (distorções de lente), o algoritmo obteve bons resultados. Isto é comprovado através dos testes feitos utilizando uma "*webcam*".

Capítulo 8

8 Conclusões Gerais e Trabalhos Futuros

Este capítulo apresenta algumas conclusões gerais e um sumário das conclusões consideradas principais de cada passo do algoritmo, visto que estas foram apresentadas no final de cada capítulo correspondente. Após as conclusões, são enumeradas sugestões de trabalhos futuros, algumas delas referentes a possíveis melhorias no algoritmo ou adaptações para casos em que o algoritmo possa falhar, como a presença de iluminação e sombra em algumas partes do modelo numa mesma imagem.

8.1 Conclusões

O algoritmo de acompanhamento de cenas seguido de calibração de câmeras proposto nesta tese gerou bons resultados mesmo quando aplicado a imagens com ruídos provenientes de uma transmissão de televisão. O objetivo de obter um algoritmo que pudesse ser usado em diversos computadores disponíveis foi alcançado. Nos computadores onde foram realizados testes, os tempos de processamento empregados foram bem abaixo do tempo necessário para o processamento em tempo real, apesar do algoritmo envolver vários passos. O tempo restante poderia ser usado, por exemplo, para desenhar anúncios e logotipos juntamente com a visualização do modelo reconstruído.

Apesar do problema apresentado nesta tese poder ser considerado complexo, o algoritmo proposto é resultado de uma seqüência de passos de complexidade menor, alguns

deles bem conhecidos na literatura e outros desenvolvidos no âmbito desta pesquisa para atender à questão do desempenho.

Os resultados obtidos demonstram o bom desempenho do método nos casos em que a seqüência de imagens mostra a maior parte das linhas do modelo com razoável nitidez. O algoritmo é robusto quanto à oclusão parcial de algumas das linhas, como mostrado na Seção 7.4. A sensibilidade a oclusão é mais séria na primeira imagem da seqüência. A partir da segunda imagem, pode-se obter bons resultados mesmo com várias linhas ocultas (é suficiente que sejam mantidas quatro retas formando um quadrilátero). Outro fator que pode causar a falha do algoritmo é a existência de sombra parcial na região em que aparecem as linhas do modelo.

A seguir, apresentamos um sumário das principais conclusões de cada passo:

- O realce dos pontos das imagens pertencentes às linhas é melhor feito através da aplicação do filtro laplaciano da gaussiana no negativo da luminância com uma segmentação por limiar.
- A estratégia de dividir a imagem em células, proposta nesta tese e apresentada no Capítulo 3, resolve bem o problema da extração de segmentos de retas quando a imagem contém regiões com características diversas, como regiões com muitos pontos com intensidade alta depois da imagem passar pela filtragem descrita no item anterior, portanto não possuindo linhas passando por esses pontos e possuindo regiões com poucos pontos dispostos sobre uma linha.
- Uma boa maneira para identificar linhas do modelo na imagem é a utilização do método baseado na árvore de interpretações. Como os modelos de interesse nesta tese são compostos de linhas paralelas a uma das duas direções mutuamente ortogonais, um conjunto de apenas 5 restrições, citadas na Seção 4.2.3, é suficiente.
- A formulação descrita na Seção 5.1 para encontrar uma homografia permitindo utilizar pontos impróprios exige um número menor de pontos. Por outro lado, a outra formulação, que utiliza apenas pontos próprios e baseia-se em minimizar o erro geométrico, citada na mesma seção, pode apresentar melhores resultados.
- O reajuste das linhas utilizando uma faixa de tolerância e operando com a imagem filtrada e segmentada, conforme o primeiro item deste sumário, apresentou bons

resultados para a nova localização dos segmentos na seqüência de quadros. Uma estimativa de localização de segmentos baseada em quadros passados é importante, podendo conduzir a resultados melhores.

8.2 Trabalhos Futuros

Os trabalhos futuros estão divididos em duas categorias em subseções separadas: na primeira, são apresentadas sugestões para possíveis aperfeiçoamentos no algoritmo como um todo ou em um passo específico; na segunda, são descritas aplicações, em especial uma muito utilizada pelas emissoras de televisão e que pode ser adaptada usando o algoritmo proposto nesta tese.

8.2.1 Aprimoramentos

Como foi mencionado ao longo desta tese, o algoritmo proposto está dividido em passos com razoável independência. Isto significa que um passo pode ter seu método modificado sem a necessidade de se reescreverem os demais passos. Portanto, são listadas aqui algumas sugestões aplicáveis a determinados passos apenas.

Com relação aos passos I e VIII, existe um parâmetro importante na segmentação usando o método de segmentação por limiar ou valor de corte. Foram feitas tentativas de obter automaticamente valores para tal parâmetro, como visto no Capítulo 2, utilizando o método de Otsu. Essas tentativas funcionaram bem em muitos casos, porém o método falha quando o histograma de cores da imagem não apresenta duas modas bem definidas. Esta falha ocorre principalmente quando parte da cena está iluminada e outra parte com sombra. Outra tentativa foi apresentada na Seção 6.4 através de uma heurística baseada no gráfico que apresenta o número de segmentos que consegue-se extrair dado um valor para o limiar de corte. A heurística executa o algoritmo proposto por inteiro para cada valor de limiar. Como o tempo de processamento é bem baixo, a determinação do limiar de corte através dessa heurística é quase que imediato e apresenta bons resultados na prática. Um trabalho futuro pode pesquisar métodos para obter pelo menos uma estimativa para este parâmetro utilizando algoritmos adaptativos a partir da divisão da imagem em regiões. Uma melhora no passos I e VIII acarretaria uma maior eficácia dos passos II e VI. No passo I, poder-se-ia realizar a segmentação em conjunto também com as cores dos pontos da imagem e não apenas

baseando-se na filtragem *LoG*. No passo VIII, pode-se tentar aumentar a performance realizando a filtragem apenas em regiões do interior das faixas de tolerância usadas no passo VI e também explorando as direções dessas faixas para melhorar a eficiência do filtro.

Outro item que pode ser explorado está relacionado com a utilização dos valores (ou intensidades) dos pontos da imagem resultante do filtro *LoG* como pesos no método de mínimos quadrados para obterem-se os segmentos. Poderíamos tentar utilizar pesos obtidos através de uma função que destaque as intensidades dos pontos de interesse, o que nos permitiria relaxar ou até tentar eliminar a etapa de segmentação do passo I, utilizando diretamente a imagem resultante da filtragem.

Em relação ao passo III, responsável pelo reconhecimento dos segmentos de retas extraídos relacionados com segmentos do modelo, pode-se tentar ajustar o critério de paralelismo dado na equação (4.2) de forma a obter os parâmetros η e ρ melhores para resolver problemas como os ilustrados na Figura 4.18. Com este ajuste, seria possível buscar meios de estender o modelo de segmentos dispostos apenas em duas direções (ou perpendiculares entre si ou paralelos) para mais direções, como diagonais. Ainda no passo III, pode-se tentar usar as junções (formadas pelas interseções das linhas do modelo) para acelerar as interpretações e técnicas de "branch and bound" para melhorar a poda da árvore de interpretação. Pode-se também buscar outros métodos capaz de reconhecer padrões contendo outras figuras geométricas diferentes de segmentos de retas, tais como círculos e elipses. Pode-se tentar trabalhar com diversos modelos ao mesmo tempo, fazendo que o método reconheça por meios de *templates* quais modelos estão na cena e presentes na imagem.

Nos passos IV e VII, encontramos transformações projetivas planares. Pode-se buscar métodos que encontrem essas transformações minimizando um determinado erro de forma a obter melhores soluções, ou então encontrar as transformações diretamente a partir de segmentos ou linhas.

Em relação ao reajuste das linhas, um coletor de pontos baseado nas faixas de tolerância pode ser aprimorado para realizar a tarefa mais rapidamente. Pode-se, também, considerar a utilização de outra forma de estimativa de localização de reajuste baseada em quadros passados, podendo ser empregados três ou mais quadros anteriores.

Como foi mencionado no final da Seção 6.3, caso não for possível reconstruir o modelo quando estiver processando um quadro de uma seqüência, o algoritmo deve ser

reiniciado e um fator que pode ocasionar isto é a existência de corte seco. Um trabalho interessante é detectar a existência dos cortes secos previamente [Silva98] e incorporar ao algoritmo.

Embora o algoritmo apresentado seja capaz de realizar calibrações de câmeras em tempo real para uma seqüência de imagens, a seqüência resultante da inserção de novos elementos na cena sofre alguns "graus de nervosismo", devido a flutuações na posição e orientação das câmeras. Futuramente, pode-se investigar processos para suavizar a trajetória das câmeras através da aplicação do filtro de Kalman ([Welch+]) ou outras técnicas.

Um trabalho também interessante consiste em desenvolver técnicas de acompanhamento de outros objetos que se movem sobre o modelo, como por exemplo jogadores e bola sobre um campo de futebol.

Para aprimorar a técnica de acompanhamento, a partir da segunda imagem de uma seqüência, quando não for possível realizar o reajuste de pelo menos 4 linhas conforme descrito na Seção 7.4, pode-se incorporar algoritmos mais gerais de *tracking*, encontrados na literatura, como mencionado na Seção 1.4.

Finalmente, pode-se investigar um algoritmo eficiente para desenhar objetos no campo de futebol por detrás dos jogadores, dando a impressão de que os objetos desenhados estão ao nível do gramado e os jogadores estão andando sobre eles.

8.2.2 Aplicações

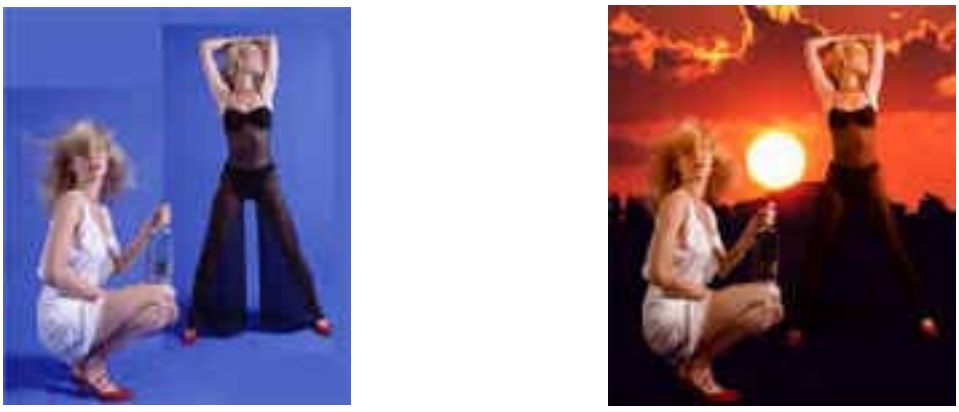
Existem diversas aplicações na área de visão computacional relativo a calibração de câmeras. Algumas são:

- Reconstruir a cena tridimensional: com a câmera calibrada, pode-se estimar a posição de um objeto, dada sua posição na imagem, através de um processo inverso da visualização. Um problema ocorre pelo fato dessa inversa não ser um ponto no mundo tridimensional mas sim uma reta, mas em algumas aplicações pode fixar uma coordenada para este ponto, como é o caso de uma partida de futebol, no qual, podemos assumir que todos os jogadores estão no chão em um instante momento, portanto, sua coordenada z é igual a zero.
- Sobreposição de imagens: muitas vezes queremos realçar um objeto na imagem que conhecemos sua posição no mundo, mas que por algum motivo, ele não

aparece legível ou então é sobreposto, parcialmente ou totalmente, por algum outro objeto. Isto é muito utilizado em sistema de realidade aumentada, onde com um óculos especial contendo uma tela de cristal liquido, uma pessoa consegue ver um objeto real e um outro artificial na frente, para por exemplo, auxiliar no conserto de uma peça mecânica.

- Observar a cena tridimensional de um outro ponto de vista: este é o objetivo principal do sistema Juiz Virtual. O usuário pode verificar visualmente se um determinado jogador estava impedido em um lance a partir de um outro ponto de vista.
- Realizar medidas: para realizar medida no mundo tridimensional a partir de pontos da imagem, basta utilizar a mesma técnica para reconstruir a cena tridimensional. Para medir distância entre dois pontos, realizada a "inversa" da projeção para cada ponto e mede a distância desses pontos resultantes no mundo tridimensional.

Outra aplicação interessante está relativo a cenários virtuais. Hoje em dia, é muito comum assistir um programa de televisão e observar que um cenário virtual foi colocado na transmissão através de técnicas baseadas em "*chroma key*" como ilustrados na Figura 8.1 e na Figura 8.2. As duas cores mais utilizadas para estas técnicas são azul ou verde. Na Figura 8.1 apenas foi substituído o azul do fundo por uma imagem sem tratar sombras. Já na Figura 8.2 foi utilizado um recurso de sombreamento, como pode ser visto, observado que há uma sombra do cachorro nas imagens dos cenários virtuais – Figura 8.2(b) e Figura 8.2(c) – conforme a sombra obtida no cenário real – Figura 8.2(a).



(a) (b)

Figura 8.1 – Exemplo de aplicação que utiliza "*chroma key*" – fundo azul.

(a)

(b)

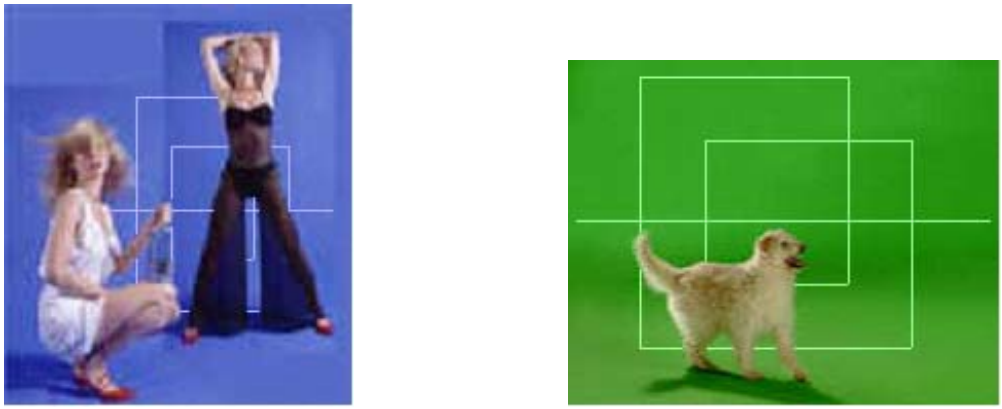
(c)

Figura 8.2 – Exemplo de aplicação que utiliza "*chroma key*" – fundo verde.

O problema de substituição surge quando se quer que a visualização do cenário virtual seja modificada conforme a câmera do cenário real. Uma solução adotada é a câmera possuir sensores capaz de informar ao sistema responsável pelo casamento dos dois cenários seus parâmetros intrínsecos e extrínsecos. O maior problema dessa solução é a restrição à esta câmera e o custo de uma câmera com sensores pode ser alto.

Outra solução possível é utilizar um padrão junto ao fundo azul ou verde e empregar as técnicas apresentadas nesta tese para realizar um acompanhamento de cenário ou câmera. A principal vantagem é poder aplicar esta técnica com qualquer câmera, sem a necessidade de utilizar nenhum equipamento específico. Um padrão para esta técnica pode ser a imagem ilustrada na Figura 7.14 e sua inserção nos fundos das figuras 8.1(a) e 8.2(a) é ilustrada na Figura 8.3. A escolha desse modelo deve-se ao fato dele não apresentar simetria para que se evite obter mais de uma solução quando é aplicado o método de reconhecimento baseado em um modelo. Esta solução pode ser adotada em virtude dos bons resultados obtidos quando uma pessoa cobriu parte do modelo e mesmo assim o sistema o localizou, como mostrado na

Figura 7.11. Pode-se também investigar um outro padrão ou até trabalhar com mais de um ao mesmo tempo.



(a) (b)

Figura 8.3 – Inserção do padrão no plano de fundo.

Referências

[Abdel-Aziz+71] Abdel-Aziz, Y.I., Karara, H.M., "*Direct Linear Transformation into Object Space Coordinates in Close-Range Photogrammetry*", Symposium on Close-Range Photogrammetry, University of Illinois at Urbana-Champaign, Urbana, Illinois, January 26-29, pp. 1-18, 1971.

[Abdel-Aziz+74] Abdel-Aziz, Y.I., Karara, H.M., "*Photogrammetric Potential of Non-Metric Cameras*", Civil Engineering Studies, Photogrammetry Series No. 36, University of Illinois at Urbana-Champaign, Urbana, Illinois, March, 1974.

[Alvarez01] Alvarez, B.S.V., "Edição tridimensional de fotografias arquitetônicas", Dissertação de Mestrado, Departamento de Informática, PUC-Rio, agosto, 2001.

[Andrews91] Andrews, D.W.K., "*Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimation*", Econometrica, Vol. 59, No.3, pp. 817-858, 1991.

[Araújo+98] Araújo, H., Carceroni, R.L., Brown, C.M., "*A Fully Projective Formulation to Improve the Accuracy of Lowe's Pose-Estimation Algorithm*", Computer Vision and Image Understanding, vol. 70, no. 2, pp. 227-238, 1998.

[Armstrong+95] Armstrong, M., Zisserman, A., "*Robust Object Tracking*", In Proc. Asian Conference on Computer Vision, volume I, pages 58-61, 1995.

[Babaud+86] Babaud, J., Witkin, A., Baudin, Duda, R., "*Uniqueness of the Gaussian Kernel for Scale-Space Filtering*", IEEE Trans. on PAMI, 8, January 1986.

[Bebie+00] Bebie, T., Bieri, H., "*A Video-Based 3D-Reconstruction of Soccer Games*", In M. Gross and F.R.A. Hopgood, editors, Proc. EUROGRAPHICS'2000, 19(3), 1996.

[Bellon+90] Bellon, O.R.P., Tozzi, C.L., "*Estudo dos Parâmetros na Calibração de uma Câmera CCD: Uma Abordagem Prática*", III Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens, SIBGRAPI'90, págs. 264-273, 30 de Maio a 1 de Junho, Gramado, RS, 1990.

[Blake+93] Blake, A., Curwen, R., Zisserman, A., "*A Framework for Spatiotemporal Control in the Tracking of Visual Contours*", International Journal of Computer Vision, 11(2):127-146, October, 1993.

[Brown71] Brown, D.C., "*Close-Range Camera Calibration*", Photogrammetric Engineering, Vol. 37, No. 8, pp. 855-866, 1971.

[Canny86] Canny, J., "*A Computational Approach to Edge Detection*" IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-8, no. 6, pp. 679--698, November 1986.

[Carvalho+98] Carvalho, P.C.P., Szenberg, F., Gattass, M, "*Image-Based Modeling Using a Two-Step Camera Calibration Method*", Proceedings of International Symposium on Computer Graphics, Image Processing and Vision, SIBGRAPI'98, October 20-23, pp. 388-395, 1998.

[Carvalho+99] Carvalho, P.C.P., Figueiredo, L.H., Gomes, J., Velho, L., "*Métodos de Otimização em Computação Gráfica*", 22º Colóquio Brasileiro de Matemática, IMPA, 26 à 30 de julho, 1999.

[Clarke+96] Clarke, J.C., Carlsson, S., Zisserman, A., "*Detecting and Tracking Linear Features Efficiently*", In R. B. Fisher and E. Trucco, editors, Proc. 7th British Machine Vision Conference, Edinburgh, pages 415-424, 1996.

[Clarke+96a] Clarke, J.C., Zisserman, A., "*Detection and Tracking of Independent Motion*", Image and Vision Computing, vol. 14, pp. 565-572, August, 1996.

[Debevec+96] Debeved, P.E., Taylor, C.J., Malik, J., "*Modeling and Rendering Architecture from Photographs: A Hybrid geometry- and image-base approach*", Computer Graphics Proceedings, SIGGRAPH'96, pp. 11-31, New Orleans, 1996.

[DigitalReplay] Digital Replay system, http://www.orad.co.il/sport/Pages/dreplay.htm.

[Duda+72] Duda, R.O., Hart, P.E., "*Use of Hough Transform to Detect Lines and Curves in Picture*", Communications of the ACM, 15(1):11-15, 1972.

[Faig75] Faig, W., "*Calibration of Close-Range Photogrammetry Systems: Mathematical Formulation*", Photogrammetric Engineering and Remote Sensing, Vol. 41, No. 12, pp. 1479-1486, 1975.

[Faugeras93] Faugeras, O., "*Three-Dimensional Computer Vision: a Geometric ViewPoint*", MIT Press, 1993.

[Firstandten] First and Ten system, http://www.sportvision.com/how_it_works/first.asp.

[Fisher94] Fisher, R.B., "*Performance Comparison of Ten Variations on the Interpretation-Tree Matching Algorithm*", Proc. 1994 European Conference on Computer Vision, ECCV'94, Stockholm, Sweden, pp 507-512, 1994.

[Foley+95] Foley, J. D., Van Dam, A., Feiner, S. K., Huhes, J. F., "*Computer Graphics: Principles and Practices*", Systems Programming, 2nd edition in C, Addison-Wesley, 1995.

[Ganapathy84] Ganapathy, S., "*Decomposition of Transformation Matrices for Robot Vision*", Proceedings of International Conference on Robotics and Automation, pp. 130-139, 1984.

[Gennery79] Gennery, D.B., "*Stereo-Camera Calibration*", Proceedings of Image Understandings Workshop, November, pp. 101-108, 1979.

[GolfProVision] GolfProVision system, http://www.questec.com/q2001/prod_gf.htm.

[Gomes+97] Gomes, J., Velho, L., "*Image Processing for Computer Graphics*", Springer-Verlag, 1997.

[Gonzales+93] Gonzales, R.C., Woods, R.E., "*Digital Image Processing*", Addison-Wesley Publishing Company, September 1993.

[Grimson90] Grimson, W.E.L., "*Object Recognition by Computer: The Role of Geometric Constraints*", Massachusetts Institute of Technology, 1990.

[Hall+82] Hall, E.L., Tio, M.B.K., McPherson, C.A., Sadjadi, F.A., "*Curved Surface Measurement and Recognition for Robot Vision*", Conference

Record, IEEE Workshop on Industrial Application of Machine Vision, May 3-5, 1982.

[Hough62] P.V.C. Hough, "*Method and Means of Recognizing Complex Patterns*", U.S. Patent 3069654, 1962.

[IUP] IUP - Portable User Interface, http://www.tecgraf.puc-rio.br/iup.

[Intille+94] Intille, S.S., Bobick, A.F., "*Tracking Using a Local Closed-World Assumption: Tracking in the Football Domain*", Master Thesis, M.I.T. Media Lab, 1994

[Isaguirre+85] Isaguirre, A., Pu, P., Summers, J., "*A New Development in Camera Calibration: Calibrating A Pair of Mobile Camera*", Proceedings of International Conference on Robotics and Automation, pp. 74-79, 1985.

[Jain+95] Jain, R., Kasturi, R., Schunck, B.G., "*Machine Vision*", McGraw-Hill, 1995.

[JuizVirtual] Sistema Juiz Virtual, http://www.tecgraf.puc-rio.br/juizvirtual.

[Karara79] Karara, H.M., "*Handbook of Non-Topographic Photogrammetry*", American Society of Photogrammetry, 1979.

[Kasprzak+94] Kasprzak, W., Niemann, H., "*Moving Segment Detection in Monocular Image Sequences under Egomotion*", In Signal Processing VII : Theories and Applications, volume 2, pp. 708-711. EUSIP Association, Lausanne, CH, 1994.

[Kim+98] Kim, T., Seo, Y., Hong, K.-S., "*Physics-based 3D position analysis of a soccer ball from monocular image sequences*", International Conference on Computer Vision, Bombay, India, pp. 721-726, January, 1998,

[Kim+98a] Kim, T., Seo, Y., Hong, K.-S., "*Soccer game analysis: 3D positioning of a soccer ball from single view*", Frontiers on Computer Vision, Tokyo, February, 1998

[Kolb+95] Kolb, C., Mitchell, D., Hanrahan, P., "*A Realistic Camera Model for Computer Graphics*", Computer Graphics Proceedings, SIGGRAPH'95, pp. 317-324, Los Angeles, 1995.

[Lee+95] Lee, T.C.M., Talbot, H., "*A Fast Method for Detecting and Matching Linear Features in Images*" in Proceedings of DICTA 95, Digital Image Computing: Techniques and Applications, 1995.

[Lee+97] Lee, T.C.M., Talbot, H., "*Automatic Reconnection of Linear Segments by the Minimum Description Length Principle*", Proceedings of DICTA-97, Digital Image Computing: Techniques and Applications, Auckland, pp. 555-560, 1997.

[Li+94] Li, M., Betsis, D., L., J.M., "*Kinametic Calibration of the KTH Head-Eye System*", CVAP171, Computational Vision & Active Perception Laboratory, Department of Numerical Analysis and Computing Science, Stockholm, Sweden, November, 1994.

[Li+95] Li, M., L., J.M., "*Some Aspects of Zoom-Lens Camera Calibration*", CVAP172, Computational Vision & Active Perception Laboratory, Department of Numerical Analysis and Computing Science, Stockholm, Sweden, February, 1995.

[Li+96] Li, M., L., J.M., "*Some Aspects of Zoom-Lens Camera Calibration*", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18, No. 11, November, 1996.

[Lowe91] Lowe, D.G., "*Fitting Parameterized 3-D Models to Images*", IEEE Transactions on Pattern Analysis and Machine Intelligence, 13, 5, pp. 441-450, May, 1991.

[Malhotra71] Malhotra, "*A Computer Program for the Calibration of Close-Range Cameras*", Proceedings of Symposium on Close Range Photogrammetric Systems, Urbana, Illinois, 1971.

[Martins+81] Martins, H.A., Birk, J.R., Kelley, R.B., "*Camera Models Based on Data from Two Calibration Planes*", Computer Graphics and Image Processing, 17, pp. 173-180, 1981.

[Mendonça+99] Mendonça, P.R.S., Cipolla, R., "*A Simple Technique for Self-Calibration*", Computer Vision and Pattern Recognition, pp. 23-25, Fort Collins, Colorado, June, 1999.

[Nitzberg+92] Nitzberg, M., Shiota, T., "*Nonlinear Image Filtering with Edge and Corner Enhancement*", IEEE Transactions on Pattern Analysis and Machine Intelligence, 14(8):826-833, 1992.

[Okamoto81] Okamoto, A., "*Orientation and Construction of Models, Part I: The Orientation Problem in Close-Range Photogrammetry*", Photogrammetric Engineering and Remote Sensing, Vol. 47, No. 10, pp. 1437-1454, 1981.

[Okamoto84] Okamoto, A., "*The Model Construction Problem Using the Collinearity Condition*", Photogrammetric Engineering and Remote Sensing, Vol. L, No. 6, pp. 705-711, 1984.

[Olson94] Olson, C. F., "*Probabilistic Indexing: A New Method of Indexing 3D Model Data from 2D Image Data*", In Proceedings of the Second CAD-Based Vision Workshop, pp. 2-8, 1994.

[ORAD] ORAD HI-TEC SYSTEMS LTD, http://www.orad.co.il.

[Otsu79] Otsu, N., "*A threshold selection method from gray-level histograms*", IEEE Transaction Systems Man Cybernet. 9(1):62-69, 1979.

[Pernuš+94] Pernuš, F., Leonardis, A., Kovačič, S., "*Two-Dimensional Object Recognition Using Multiresolution Non-Information-Preserving Shape Features*", Pattern Recognition Letters, 11, pages 1071-1079, 1994.

[Perona+90] Perona, P., Malik, J., "*Scale-Space and Edge Detection Using Anisotropic Diffusion*", IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(7):629-639, 1990.

[Perš+00] Perš, J., Kovačič, S., "*Computer Vision System for Tracking Players in Sports Games*", Proceedings of the First International Workshop on Image and Signal Processing and Analysis IWISPA 2000, pp. 81-86, Pula, Croatia, June 14-15. 2000.

[Perš+01] Perš, J., Bon, M., Kovačič, S., "*Errors and Mistakes in Automated Players Tracking*", Proceedings of Sixth Computer Vision Winter Workshop, Bled, Slovenia, pp. 25-36, February 7-9. 2001.

[PicthTrax] PicthTrax system, http://www.questec.com/q2001/prod_pt.htm.

[Rahimi+01] Rahimi, A., Morency, L.P., Darrell, T., "*Reducing Drift in Parametric Motion Tracking*", ICCV2001, June 2001.

[Rousso+96] Rousso, B., Avidan, S., Shashua, A., Peleg, S., "*Robust Recovery of Camera Rotation from Three Frames*", In Conference on Computer Vision and Pattern Recognition, CVPR 1996, June 18-20, San Francisco, Ca., 1996.

[Schweitzer+98] Schweitzer, H., Kulkarni, S., "*Computational Limitations of Model-Based Recognition*", International Journal of Intelligent Systems, 13(5):431-443, 1998.

[Seo+97] Seo, Y., Choi, S., Kim, H., Hong, K.-S., "*Where Are the Ball and Players?: Soccer Game Analysis with Color-based Tracking and Image Mosaic*", In proc. IAPR International Conference on Image Analysis and Processing, pages 196-203, Florence, Italy, 1997.

[Shull99] Shull, Jim.The Beginner's Guide to Pinhole Photography. Amherst Media. Inc., ISBN: 0-936-26270-2, 1999.

[Silva98] Silva, R.J., "*Processamento Geométrico de Objetos Gráficos Volumétricos*", Tese de Doutorado, IMPA, julho, 1998.

[Sobel74] Sobel, I., "*On Calibration Computer Controlled Cameras for Perceiving 3-D Scenes*", Artificial Intelligence, 5, pp. 185-198, 1974.

[Spiess+96] Spiess, S., Li, M., "*Kinematic Calibration of an Active Binocular Head for Online Computation of Epipolar Geometry*", CVAP205, Computational Vision & Active Perception Laboratory, Department of Numerical Analysis and Computing Science, Stockholm, Sweden, October, 1996.

[State+96] State, A., Hirota, G., Chen, D.T., Garrett, W.F., Livingston, M.A, "*Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking*", In Holly Rushmeier, editor, SIGGRAPH 96 Conference Proceedings (New Orleans, LO), Annual Conference Series, pages 429-438. ACM SIGGRAPH, Addison Wesley, August 1996.

[Strat84] Strat, T.M., "*Recovering the Camera Parameters from a Transformation Matrix*", Proceedings: DARPA Image Understanding Workshop, October, pp. 264-271, 1984.

[Sutherland74] Sutherland, I., "*Three-Dimensional Data Input by Tablet*", Proceedings of the IEEE, Vol. 62, No. 4, April, pp. 453-461, 1974.

[Svedberg+99] Svedberg, D., Stefan, C., "*Calibration, Pose and Novel Views from Single Images of Constrained Scenes*", SCIA'99, 1999.

[Symes98] Symes, P.D., "*Video Compression*", McGraw Hill, April, 1998.

[Szenberg+2001] Szenberg, F., Carvalho, P.C.P., Gattass, M., "*Automatic Camera Calibration for Image Sequences of a Football Match*", Advances in Pattern Recognition - ICAPR 2001, Second International Conference, Rio de Janeiro, Brazil, March 11-14, ISBN 3-540-41767-2, pp. 301-311, 2001.

[Tekalp95] Tekalp, A.M., "*Digital Video Processing*", Prentice Hall, August, 1995.

[TennisProVision] TennisProVision system, http://www.questec.com/q2001/prod_tn.htm.

[Tommaselli+91] Tommaselli, A.M.G., Tozzi, C.L., " *Calibração de Câmeras Usando Feições Genéricas*", Tutorial, IV Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens, SIBGRAPI'91, Julho 14-17, São Paulo, 1991.

[Tsai86] Tsai, R.Y., "*An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision*", CVPR'86 Proceeding, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Miami Beach, FL, June 22-26, pp. 364-373, 1986.

[VirtuaLive] VirtuaLive system, http://www.orad.co.il.

[VirtualReplay] Virtual Replay system, http://www.orad.co.il/sport/Pages/vreplay.htm.

[Vosselman+96] Vosselman, G., Haralick, R. M., "*Performance Analysis of Line and Circle Fitting in Digital Images*", Workshop on Performance Characteristics of Vision Algorithms, Cambridge, April 19, 1996.

[Wang+90] Wang, L.L., Tsai, W.H., "*Computing Camera Parameters Using Vanishing-Line Information from a Rectangular Parallelepiped*", Machine Vision and Applications, No. 3, 1990.

[Watt+92] Watt, A., Watt, M., "*Advanced Animation and Rendering Techniques: Theory and Practice*", ACM Press, New York, NY, 1992.

[Welch+] Welch, G, Bishop, G., "*An Introduction to the Kalman Filter*", In: http://www.cs.unc.edu/~welch/media/ps/kalman.ps.

[Witkin83] Witkin, A., "*Scale-space filtering*", In International Joint Conference on Artificial Intelligence, pages 1019-1021, 1983.

[Wong75] Wong, K.W., "*Mathematical Formulation and Digital Analysis in Close-Range Photogrammetry*", Photogrammetric Engineering and Remote Sensing, Vol. 41, No. 11, pp. 1355-1373, 1975.

[Woo+99] Woo, M., Neider, J., Davis, T., Shreiner, D., "*OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.2*", Addison-Wesley, 1999.

[Yakimovsky+78] Yakimovsky, Y., Cunningham, R., "*A System for Extracting Three-Dimensional Measurements from a Stereo Pair of TV Cameras*", Computer Graphics and Image Processing, 7, pp. 195-210, 1978.

[Zeller+96] Zeller, C., Faugeras, O., "*Camera Self-Calibration from Video Sequences: the Kruppa Equations Revised*", Institut National de Recherche en Informatique et en Automatique – INRIA Sophia Antipolis, No. 2793, France, Février, 1996.

[Ziou+98] Ziou, D., Tabbone, S., "*Edge Detection Techniques – An Overview*", International Journal of Pattern Recognition and Image Analysis, Vol. 8, No. 4, pp. 537-559, 1998.

Acompanhamento de Cenas com Calibração Automática de Câmeras

Tese de Doutorado apresentada por **Flávio Szenberg** em 19 de dezembro de 2001 ao Departamento de Informática da PUC-Rio e aprovada pela Comissão Julgadora, formada pelos seguintes Professores:

Prof. Marcelo Gattass
Orientador
Depto. de Informática

Prof. Paulo Cezar Pinto Carvalho
Co-Orientador
IMPA

Prof. Sidnei Paciornik
Depto. de Ciência de Materiais e Metalurgia – PUC-Rio

Prof. Díbio Leandro Borges
Depto. de Ciência da Computação – PUC-PR

Prof. Antonio Alberto Fernandes de Oliveira
COPPE – UFRJ

Prof. Luiz Carlos Pacheco Rodrigues Velho
IMPA

Prof. Waldemar Celes Filho
Depto. de Informática – PUC-Rio

Visto e permitida a impressão
Rio de Janeiro, 24/01/2002

Coordenador dos Programas de Pós-Graduação e
Pesquisa do Centro Técnico Científico