Visão Artificial e Realidade Mista

Mestrado em Engenharia Informática e Multimédia

Computer Vision and Mixed Reality

Project 2 – Marker
Based Augmented Reality

Rúben Santos, 49063

Índice

1 Introdução

A Computer vision é uma área onde são utilizadas técnicas computacionais para obter uma compreensão de alto nível da informação de imagens, vídeos ou outros tipos de informações visuais. Neste trabalho, foram utilizadas informações visuais para identificar e registar objetos virtuais de forma a utilizar a realidade aumentada (Augmented Reality).
O objetivo da realidade aumentada é adicionar e interagir como objeto virtuais de moda a que estes se pareçam com o mundo real, numa perspetiva do utilizador.
Na cadeira de Visão artificial e realidade mista, o segundo projeto teve como objetivos:

- Desenvolver um aplicativo de visão computacional que implemente realidade aumentada baseada em marcadores e permita a inclusão de elementos virtuais alinhados com marcadores fiduciais reais.
- Familiarização com a biblioteca OpenCV (Open-Source Computer Vision), para desenvolvimento de aplicações em tempo-real e a familiarização de uma biblioteca baseada em marcadores ArUco.

2 Calibração da câmara

No início do desenvolvimento do projeto, foi desenvolvida uma aplicação capaz de efetuar a calibração de camearas para se utilizar depois na aplicação de realidade aumentada. Esta calibração consistiu em efetuar-se a obtenção da matriz de parâmetros intrínsecos e dos coeficientes de distorção.

Neste primeiro ponto, para realizar a obtenção da matriz de parâmetros intrínsecos e dos coeficientes de distorção, foram carregadas várias fotografias (tiradas pela camara em questão a calibrar) de um checkborad. Em seguida a aplicação de calibração da camara processa cada fotografia no sentido de efetuar os cálculos necessário para obter a matriz de parâmetros intrínsecos e os coeficientes de distorção. Para finalizar a aplicação produz dois ficheiros (no formato .txt) que contem a informação da camara.

Para obter esses parâmetros intrínsecos pode-se fazer a assunção de um padrão 2D de dimensões conhecidas constituindo um plano 3D. Um dos padrões mais conhecidos e usados é o do checkerboard (tabuleiro de xadrez) pelo que foi esse o escolhido para a obtenção dos parâmetros intrínsecos das câmaras usadas no âmbito do presente projeto.

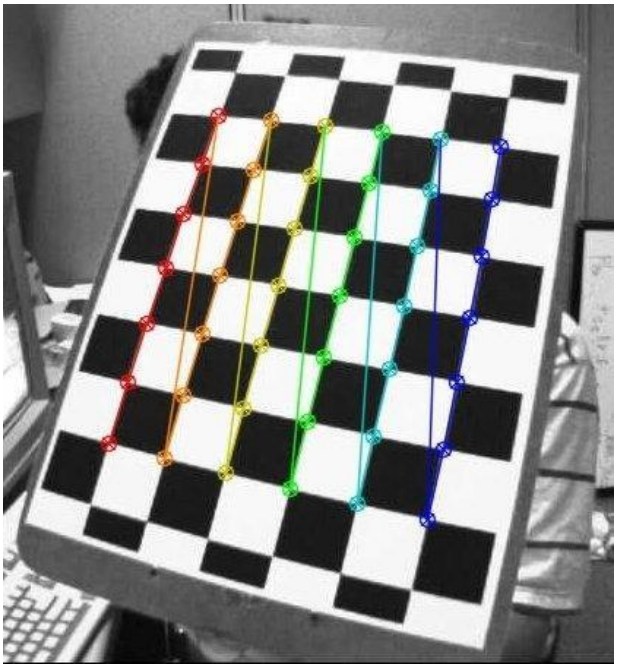


Figura 1- Exemplo de checkerboard usado em calibrações de câmaras. [Retirado de OpenCV]

Para obter estes parâmetros começa-se p0or fazer a deteção do chekerboad com recurso ao método cv.findChessboardCorners(). Ao realizar isto, todos os pontos correspondentes as interceções dos quadrados a preto estarão no plano X/Y, Z=0. (Ver resultado na figura seguinte).

Em adição, se se considerar que a unidade do referencial corresponde à dimensão do lado de um quadrado do checkerboard, conseguem-se obter as coordenadas desses pontos de acordo com o referencial considerado.

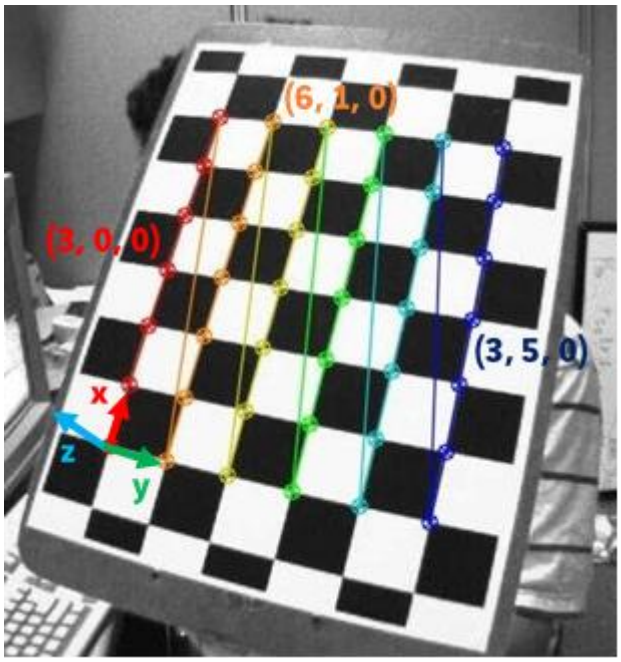


Figura 2- Pontos do checkerboard – coordenadas do mundo.

A partir daqui, usam-se ferramentas avançadas de álgebra linear com o objetivo de se calcular a matriz H e, posteriormente, separá-la em duas componentes que irão corresponder às matrizes de parâmetros intrínsecos e extrínsecos.

Para a correta realização dos cálculos dos parâmetros intrínsecos é necessário usarem-se várias imagens do checkerboard a partir de ângulos diferentes.

No final da execução da calibração, são gerados dois ficheiros .txt contendo os parâmetros intrínsecos das câmaras e os coeficientes de distorção, respetivamente.

3 Realidade aumentada baseada em marcadores

Na segunda parte do trabalho, tinha como objetivo criar uma aplicação capaz de detetar e estimar a posição estimada da camara com recurso a biblioteca ArUco com também efetuar o registo de objetos virtuais.

Para realizar este objetivo começa-se por criar um frame com a captura de vídeo, para detetar um ou mais marcadores e executar os vários estágios de renderização para que sejam exibidos os modelos ao utilizador. Na imagem seguinte podemos ver os estágios deste processo:

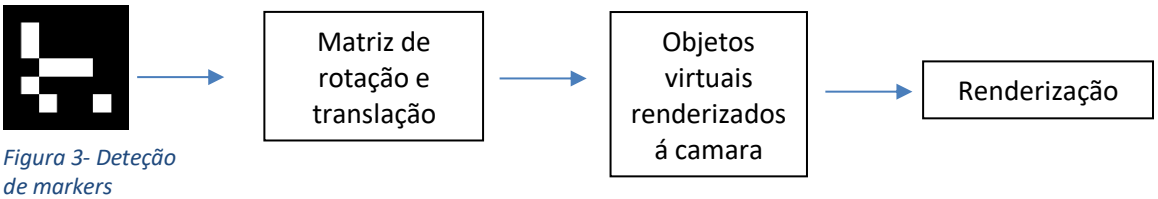


Figura 3- Deteção de markers

Neste processo, o método para detetar os markers foi o aruco.detectMarkers() que permite obter os id's de cada marker identificado na frame. Com esta informação é realizada a escolha dos objetos a renderizar (neste caso um cubo a vermelho e azul)

No passo da rotação e translação da matriz os quatro cantos do marcador servem como parâmetros para o métodos aruco.estimatePoseSingleMarkers() que devolve a pose do marcador em relação à câmara sob a forma de vetores de rotação (contendo apenas ângulos) e translação separados. Convertendo assim no primeiro passo o vetor de rotação numa matriz 3 por 3 e anexar o vetor translação de modo a perfazer [R | t].

No terceiro passo como já se obteve a matriz de parâmetros extrínsecos, resta referenciar os pontos do objeto virtual ao referencial da camara. Para isso é realizada a multiplicação dos pontos que constituem o modelo pela matriz [R| t]. Neste passo foi importante perceber que os pontos do modelo do objeto não comtemplam as coordenadas homogénea pelo que foi necessário acrescentá-la manualmente através da funcao concatenate do numpy.

Por último, tendo os três primeiros passos realizados, resta efetuar a renderização do objeto na frame.
Para "desenhar", a forma escolhida a implementar foi com base num Z-Buffer, que é uma estrutura de dados que representa a profundidade dos objetos num espaço 3D a partir da perspetiva da camara.
O método Z-buffer compara as profundidades da superfície em cada posição de pixel no plano de projeção. Normalmente o eixo z é representado como a profundidade.
O algoritmo para o método Z-Buffer começa por criar um Z-Buffer de dimensões igual ao frame inicializando-o com um valor elevado (ex.99999) e inicia-se o valor da cor para cada pixel como valor de fundo.

De seguida para cada pixel na projeção do cubo encontra-se a profundidade, ou seja, o z do cubo (x, y) correspondentes por pixel (i, j).

Por fim e para facilitar a observação do modelo, optou-se por desenha um círculo usando os pixéis restantes das projeções dos pontos 3D como centros dos mesmos, dando assim o resultado da seguinte figura:

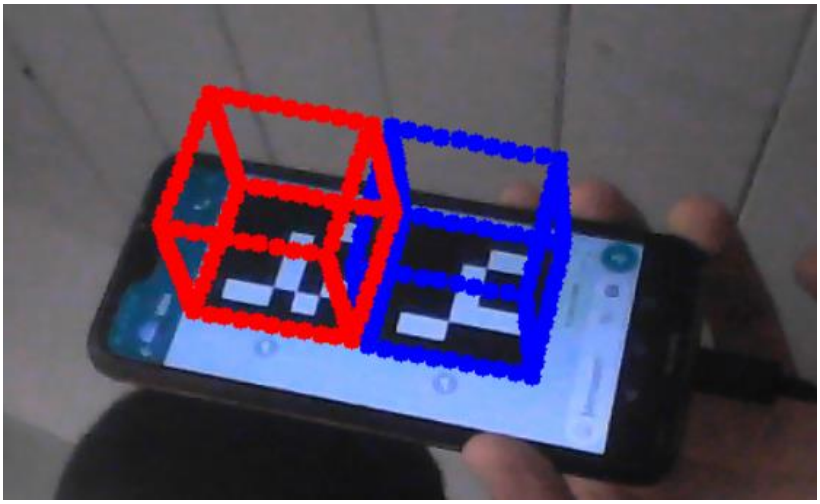
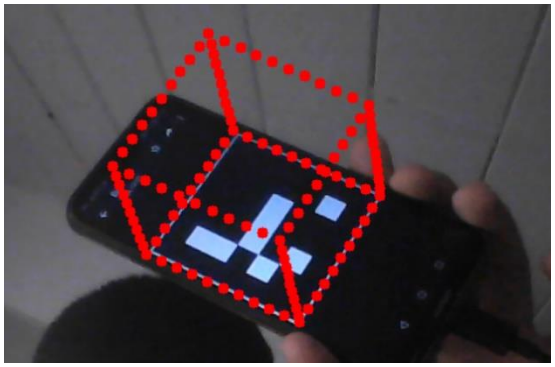
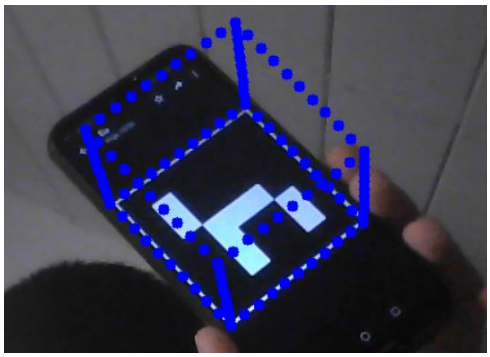


Figura 4- Renderização de objeto virtual com círculos para maior facilidade de visualização.





Ao executar a aplicação de realidade aumentada, ocorreram algumas soluções com menor aproveitamento, ou seja, ao executar a aplicação existem aos problemas na renderização de alguns pontos dos objetos virtuais e as suas projeções. Não foi adotada qualquer solução para este problema por falta de tempo e conhecimentos. O problema é demostrado visualmente na seguinte imagem:

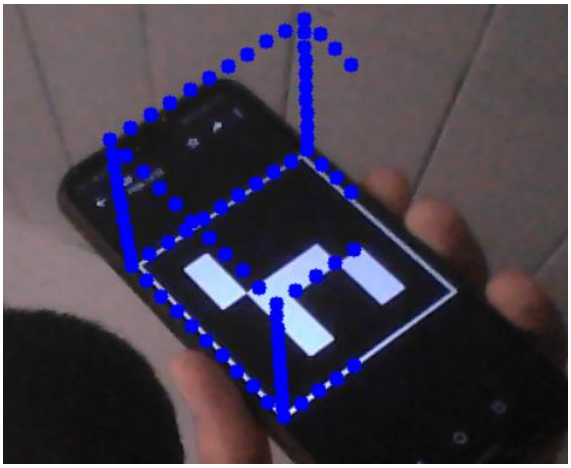


Figura 5- Problema na rederização

4 Conclusão

Para concluir, neste trabalho foram utilizadas informações visuais para identificar e registar objetos virtuais de forma a utilizar a realidade aumentada (Augmented Reality). A Realidade aumentada tem como objetivo adicionar objetos virtuais de modo que estes pareçam parte do mundo real, na perspetiva do utilizador.

No âmbito desta cadeira foram desenvolvidas aplicações de realidade aumentada com o recurso a camaras o que fez com que se obtivessem os parâmetros intrínsecos e extrínsecos da câmara. Estes parâmetros representam as transformações dos pontos 3D das coordenadas da camara no mundo e das projeções das coordenadas 3D da camara para coordenadas 2D no plano de imagem.

Foram desenvolvidas as aplicações para a calibração da camara e para a inclusão de objetos virtuais. Onde para “desenhar” os objetos virtuais se utilizou o Z-Buffer que contem as menores profundidades por pixel restantes das projeções de pontos 3D.

Por último, o único problema que se obteve neste projeto foi a falha na renderização nos pontos dos objetos virtuais quando existia um angulo e movimento entre a camara e o marcador