FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Alexandre Azevedo Marques

Estimação da pose de objectos em imagens RGB-D utilizando aprendizagem automática.

Dissertação de Mestrado Integrado em Engenharia Electrotécnica e de Computadores.

Coimbra

Setembro 2018

Estimação da pose de objectos em imagens RGB-D utilizando aprendizagem automática.

Dissertação de Mestrado Integrado em Engenharia Electrotécnica e de Computadores, ramo de especialização em Automação.

por:

Alexandre Azevedo Marques

Orientador:

Prof. Hélder de Jesus Araújo

Júri:

Prof. Hélder de Jesus Araújo

Prof. Jorge Manuel Moreira de Campos Pereira Batista

Prof. Paulo José Monteiro Peixoto

Coimbra Setembro 2018

“Sometimes science is more art than science. A lot of people dont get that."

— Rick Sanchez

Agradecimentos

Agradeço a todos os que me acompanharam até este momento da minha vida, principalmente aos meus pais pois sem o seu apoio e incentivo não teria conseguido atingir este meu objetivo.

Gostava ainda de agradecer ao meu orientador de tese Professor Hélder de Jesus Araújo, pela orientação prestada nesta última fase do meu percurso académico.

Resumo

Este trabalho foi feito no contexto duma dissertação de tese de mestrado e tem como objectivo, explorar métodos de aprendizagem automática para detecção de objectos e estimação da sua pose em imagens RGB-D .

No decorrer desta dissertação, foi implementado o método de estimação de pose utilizado em [Kehl et al., 2016], onde são extraídas características locais de uma imagem RGB-D invariantes à escala com o auxilio duma rede neuronal convolucional. Posteriormente cada amostra é comparada com uma lista de características conhecidas, de objetos sintéticos previamente amostrados, e são gerados votos relativamente aos 6 graus de liberdade dos objectos presentes na imagem em questão.

Foi ainda implementada uma variante do método com o intuito de tentar melhorar a performance dos algoritmos desenvolvidos.

Abstract

This work was developed in the context of a master thesis and it aims to explore machine learning methods for object detection and estimation of their poses in RGB-D images.

In the course of this dissertation the pose estimation method used in [Kehl et al., 2016] was implemented, where locally-sampled RGB-D patches are regressed into features with the aid of a convolutional neural network. Subsequently each sample is compared with a list of known characteristics of previously sampled synthetic objects, and are generated votes relative to the 6 degrees of freedom of the objects present in the RGB-D image.

In addition, a variant of the original method was implemented in order to try to improve its performance.

Lista de Acrónimos e Abreviaturas:

CAE *Convolutional Auto Encoder*

CNN *Convolutional Neural Network*

DOF *Degrees of Freedom*

FCTUC Faculdade de Ciência e Tecnologias da Univarsidade de Coimbra

ISR Intituto de Sistemas e Robótica

RGB *Red*, *Green and Blue*

RGB-D *Red*, *Green*, *Blue and Depth*

RNN *Recurrent Neural Network*

ROS *Robotic Operating System*

Lista de Figuras

Conteúdo

1 Introdução

A detecção de objectos tem sido tópico de investigação durante as últimas décadas, sendo crucial para múltiplas aplicações em áreas como é o caso da robótica, navegação autónoma, realidade aumentada, entre outras. A área da robótica, tende cada vez mais no sentido de tornar a interacção Robô/Humanos o mais simbiótica e natural possível, e nesse sentido é fácil imaginar cenários onde robôs são capazes de manipular objectos e auxiliar humanos em situações diversas, tais como cirurgias, construção e hotelaria ou até robôs autónomos em serviços de atendimento a clientes ou de transporte. Nestes casos conhecer a localização de um objecto no espaço 3D mostra-se insuficiente e por isso é necessário ter informação adicional acerca da pose dos objectos em questão. Uma tarefa trivial como encher um copo de água só é possível se o copo tiver uma determinada orientação.

Este tipo de problemas é um perfeito exemplo do tipo de problemas que a área de aprendizagem automática se propõe a resolver, nos últimos anos múltiplas abordagens têm sido propostas e igualado ou ultrapassado métodos clássicos e tradicionais de estado da arte. De tal modo que temos um grande interesse em explorar este novo tipo de abordagens.

1.1 Motivação

Inicialmente esta dissertação focava-se em utilizar o robô *Baxter* existente no laboratório de Visão do Instituto de Sistemas e Robótica da *FCTUC*, para manipulação de objectos dentro do seu espaço de trabalho. Para tal, o problema foi inicialmente subdividido em problemas aparentemente mais simples.

- Detecção de objectos no espaço de trabalho do *Baxter* e das suas respectivas poses.
- Planeamento da manipulação dos objectos em questão.
- Planeamento de trajetórias e controlo das articulações do robô *Baxter*.

A primeira fase focada na detecção da pose de objectos, sendo que esta deve ser robusta e capaz de reconhecer e estimar a pose de objectos conhecidos mas também para objectos "imprevistos". Este problema mostrou ser bastante complexo só por si, de tal forma que levou à reformulação do problema inicial da dissertação. Por isso, o problema passou então a ser o desenvolver um método com as características descritas em cima.

O método deve ser capaz de realizar a detecção de pose para objectos dentro dum espaço de trabalho e ser facilmente integrado em robôs como o *Baxter*, ou noutro tipo de aplicações através do *ROS*. Tendo em conta a falta de experiência e o tempo limitado, um bom objectivo seria inicialmente tentar desenvolver uma versão do método com a capacidade de detectar a pose de múltiplos objectos conhecidos, e numa fase mais final tentar generalizar o problema e detectar objectos previamente não utilizados.

2 Trabalho Relacionado

Ao longo dos anos surgiram múltiplas abordagens ao problema de detecção de objectos, variando entre si, através de factores como, o número de objectos que são capazes de detectar em simultâneo, se são dependentes de modelos sintéticos dos objectos ou se tiram partido de algoritmos de modelação, e ainda relativamente aos graus de liberdade que tentam estimar. Algumas abordagens apenas tem como objectivo detectar as projecções de objectos reais no plano de imagem (2 graus de liberdade), como por exemplo a de [Redmon et al., 2016], enquanto outras se focam em detectar os objectos no mundo real em frente à camera, podendo detectar apenas a sua posição (3 graus de liberdade) ou a sua pose (6 graus de liberdade).

Na última década, têm surgido cada vez mais métodos baseados em "*Deeplearning*". Estes tiram partido de redes neuronais complexas em conjunto com grandes *datasets* de informação relevante, como imagens préviamente classificadas, de forma a libertarem-se da necessidade em utilizar modelos prévimante conhecidos, passando assim a basearem-se na grande quantidade de amostras dos mesmo objectos. Estes tipos de abordagem tendem a alcançar performances iguais ou superiores a métodos tradicionais. Actualmente o estado da arte baseia-se fundamentalmente na exploração deste último tipo de método, visto apresentarem tendencialmente melhores resultados que métodos mais tradicionais [Kehl et al., 2016] [Redmon et al., 2016] [Ren et al., 2015] [Garon and Lalonde, 2017] [Iventosch et al., 2017].

2.1 Métodos Tradicionais

No que toca à detecção e estimação de poses de objectos, uma grande gama de trabalhos cai na categoria a que chamamos de "métodos tradicionais". Este tipo de método basea-se fundamentalmente em utilizar técnicas de visão por computador, mecanismos de filtragem, algoritmos para modelação de objectos e modelos de objectos.

Visto a dissertação ter como intuito explorar abordagens de aprendizagem automática, não vamos explorar muito este tipo de método, é aconselhada a leitura de [Yilmaz et al., 2006]

e [Luo et al., 2014] para o leitor que procure uma melhor compreensão deste tipo de abordagens.

Apesar de não ser dado um grande foco a este tipo de métodos, gostávamos de destacar um trabalho com que nos deparámos durante a fase de pesquisa. O método de [Pauwels and Kragic, 2015] é interessante porque se aproxima bastante do nosso problema inicial. Este método utiliza uma camera *Kinect RGB-D*, montada na cabeça de um robô, juntamente com n cameras RGB, muito semelhante ao nosso ambiente. No caso específico do artigo, são utilizadas $n = 2$ cameras nos $end\ efectors$ do robô, de modo a detectar e realizar *tracking* de múltiplos objectos em simultâneo, no seu espaço de trabalho e estimar as suas poses. Este método é bastante escalável em termos do número de objectos monitorizados e da quantidade de cameras utilizadas, devido aos algoritmos em que é baseado. Apesar da semelhança com o nosso problema inicial, este método não utiliza métodos de aprendizagem automática, o que é um factor de interesse nesta dissertação, no entanto, pode ser interessante para projectos futuros com o Baxter.

2.2 Métodos baseados em Aprendizagem Automática

Tem surgido cada vez mais interesse em explorar métodos baseados em abordagens de aprendizagem automática, no que toca a detecção e reconhecimento de objectos em imagens. Este tipo de abordagem tira proveito da capacidade de redes neuronais complexas, em conjunto com grandes datasets de informação relevante, de modo a conseguirem atingir boas performances no mais variado tipo de tarefas.

Há uma vasta gama de artigos na literatura sobre este tipo de métodos e as múltiplas abordagens propostas, que variam entre si relativamente ao tipo de informação que é fornecida ao sistema (imagens RGB, RGB-D, $PointClouds$,...), à arquitectura da rede e ao seu objectivo. Vamos agora falar brevemente de alguns trabalhos que surgiram durante a fase de pesquisa.

[Ren et al., 2015] É um método de localização de objectos, que tira partido de uma rede complexa, para propor regiões numa imagem RGB onde é provável haver um ou mais objectos. Posteriormente o detector desenvolvido no [Girshick, 2015], analisa e classifica de forma eficiente as regiões de objectos propostas. O resultado final consiste em múltiplas regiões na imagem ("Bounding Boxes") onde provavelmente se encontram objectos.

Ao contrário de [Ren et al., 2015], que utiliza um paço intermédio para propor regiões na imagem a um detector[Girshick, 2015]. Em [Redmon et al., 2016] foi implementada uma abordagem YOLO ("You Only Look Once"), sendo uma única rede neuronal capaz de prever regiões e a probabilidade de cada classe de objectos, tendo apenas como entrada uma imagem RGB natural. Não só conseguem bons resultados em termos de detecção como visto todo o processo ser apenas uma rede neuronal, ainda consegue atingir boas performances.

Este tipo de métodos é bastante bom a reconhecer objectos em imagens e identificá-los, no entanto a detecção tem fundamentalmente 2 graus de liberdade (o centro das regiões propostas na imagem). Outra desvantagem desta aplicação é o facto do número de classes, que a rede consegue identificar, ser fortemente dependente da arquitectura da rede, o que não é desejável para uma detecção generalizada de objectos. Como o nosso objectivo é ter previsões de 6 graus de liberdade, para objectos no ambiente em frente à camera, vamos antes abordar alguns métodos com objectivos mais semelhantes.

No método de [Iventosch et al., 2017] é proposta uma estrutura que explore uma segmentação semântica fracamente supervisionada, como parte da detecção da pose de objectos. Com auxílio de uma camera RGB, de um braço robótico e do conhecimento dos 6 graus de liberdade do seu "end-effector", [Iventosch et al., 2017] conseguem fazer de forma semi-automática a segmentação de objectos que não existiam no dataset inicial e adaptar o classificador de forma a ser capaz de detectá-los. Devido ao tempo limitado, foi optado desenvolver um método baseado apenas em imagens RGB-D, no entanto [Iventosch et al., 2017] aborda um problema idêntico ao inicialmente proposto e pode ter utilidade em projectos futuros com o Baxter.

No caso de [Garon and Lalonde, 2017] é apresentado um seguimento temporal da pose de objectos, treinando uma rede convolucional que recebe duas imagens RGB-D, uma representa a predição da pose do objecto para o frame anterior e outra o frame actual do objecto observado. Apesar de os resultados apresentados serem muito significativos, tiram partido da utilização de modelos sintéticos dos objectos em questão, conhecidos aprori, de modo a gerar dados para o treino das redes envolvidas.

Finalmente [Kehl et al., 2016] passa por uma abordagem onde amostras locais de uma imagem *RGB-D* votam independentemente numa predição para poses de possíveis objectos existentes na imagem. Cada voto é feito com base nas características de cada amostra,

extraídas com o auxilio de uma rede neuronal convolucional. Neste método treinam de raiz a rede em questão amostrando imagens *RGB-D* de objectos reais e de seguida com objectos sintéticos para os quais é conhecido o *ground truth* é gerado um arquivo onde vectores de características são associados a transformações de corpo rígido. Este arquivo é utilizado como referência de comparação para amostras de características locais reais.

Este método apresenta uma estrutura relativamente simples e permite-nos experimentar uma rede com uma arquitectura relativamente simples de forma a tentar resolver o nosso problema. Com isto em conta, decidimos seguir este trabalho como base para esta dissertação de mestrado.

3 Estruturação do método proposto

Como já foi previamente mencionado, a dissertação tem como base o método implementado por [Kehl et al., 2016], que tem como objectivo desenvolver uma abordagem que consiga realizar a detecção de objectos e as suas poses, em imagens *RGB-D* de cenas onde tanto pode haver um, nenhum ou múltiplos objectos. O método deve ainda ser capaz de ultrapassar o problema onde um ou mais objectos se encontram parcialmente ocultados.

Tendo estes requisitos em conta, optam por uma abordagem onde a cena é segmentada em amostras locais, invariantes à escala e com as camadas de cor e profundidade normalizadas, cada amostra é associada a um sistema de votos de forma a colmatar o problema de oclusões parciais. De acordo com a equação 3.1 a dimensão de cada amostra é calculada tendo em conta o seu tamanho métrico real (m), a profundidade do seu pixel central (z) e a distância focal da camera (f). No nosso caso foi considerado $m = 5cm$. Afirmame ainda que o processo de normalização das amostras evita a necessidade de lidar com problemas de modelação do ambiente.

$$Tamanho_{Amostra} = \frac{m}{z} \times f \tag{3.1}$$

De modo a avaliarem de forma relevante a importância de cada amostra, e por consequente a confiança de cada voto, foram treinadas de origem múltiplas arquitecturas de Rede Neuronais *Autoencoder*. Os *Autoencoders* são um tipo específico de arquitectura para redes neuronais, que é tipicamente utilizado para aprender, de forma eficiente e não supervisionada, uma codificação para a informação fornecia às suas entradas. Esta nova representação da entrada é tipicamente uma aproximação fiável da informação fornecida, num espaço de dimensão inferior. No caso especifico deste método são consideradas múltiplas configurações variando nomeadamente o número de camadas, as dimensões da informação codificada e o tipo de redes, nomeadamente convolucionais e perceptrão multi-camada. A configuração que apresentou melhores resultados foi uma rede neuronal convolucional *autoencoder* (CAE) capaz de codificar as amostras locais de dimenção $[32 \times 32 \times 4]$ num vector de características

relevantes com dimensão $[1 \times 256]$ e fazer uma boa reconstrução das entradas, a partir do mesmo, tendo em conta a redução de dimensionalidade que foi aplicada. Esta configuração é abordada e mais detalhe mais à frente na dissertação.

Com o extractor de características CAE treinado, a ideia passa por gerar uma lista (*codebook*) com um grande número de amostras associadas a transformações de corpo rígido, entre a pose do seu pixel central e a pose dos objectos amostrados, para ser utilizada como referência de comparação para vectores de características extraídos a partir de amostras reais. Para tal consideram um dataset com modelos sintéticos de objectos onde são conhecidas todas as informações de pose e a sua relação espacial relativa à camera e por consequente às imagens *RGB-D* simuladas. É assumido que uma amostra real tem uma relação espacial entre o seu píxel central e o centro de massa do objecto amostrado idêntica a uma registada no *codebook* com características suficientemente semelhantes. De forma a avaliar esta semelhançaé tida em conta a distância euclidiana entre amostras, no espaço de características, considerando que quanto menor for esta distância mais semelhantes são as amostras.

O processo de votação e filtragem passa por realizar uma pesquisa dos "K vizinhos mais próximos"($K - NN$) no *codebook*, para cada amostra real obtida. A pesquisa $K - NN$ devolve os K candidatos mais semelhantes a cada amostra e de seguida são avaliadas as distâncias de cada candidato à amostra em questão, sendo descartados todos os que estejam mais longe do que um limiar previamente definido. Desta forma há sempre a possibilidade de amostras muito diferentes das existentes na lista não gerarem votos, o que pode ser desejável por exemplo com amostras do ambiente. Os candidatos mais semelhantes são de seguida projectados para uma grelha, com cada célula equivalente a uma janela de 5×5 píxeis. A confiança de cada voto é obtida a partir da distância euclidiana como descrito na equação 3.2 e a confiança de cada célula é a confiança acumulada de todos os votos que nela foram projectados como na equação 3.3.

$$Vote_{Trust} = e^{-EuclideanDistance} \tag{3.2}$$

$$Cell_{Trust} = \sum Vote_{Trust} \tag{3.3}$$

É feita mais uma etapa de filtragem, desta vez descartando as células e os votos correspondentes às mesmas, que tenham uma confiança acumulada e um numero de votos inferior a limiares previamente definidos. Ao espaço de confianças final, é aplicada uma interpolação

bilinear e posteriormente são detectados os seus máximos locais. Aos votos presentes dentro das células correspondentes aos máximos, é aplicado um operador de $meanshift$ à sua componente de posição e orientação de forma independente. Este passo pode ser visto como, a extracção do ponto central dum aglomerado de pontos no espaço, no caso da componente de posição pontos no espaço cartesiano da camera [x, y, z], e no caso da orientação pontos no espaço dos ângulos de euler [raw, $pitch$, Yaw]. As predições de pose são obtidas combinando os resultados desta última operação.

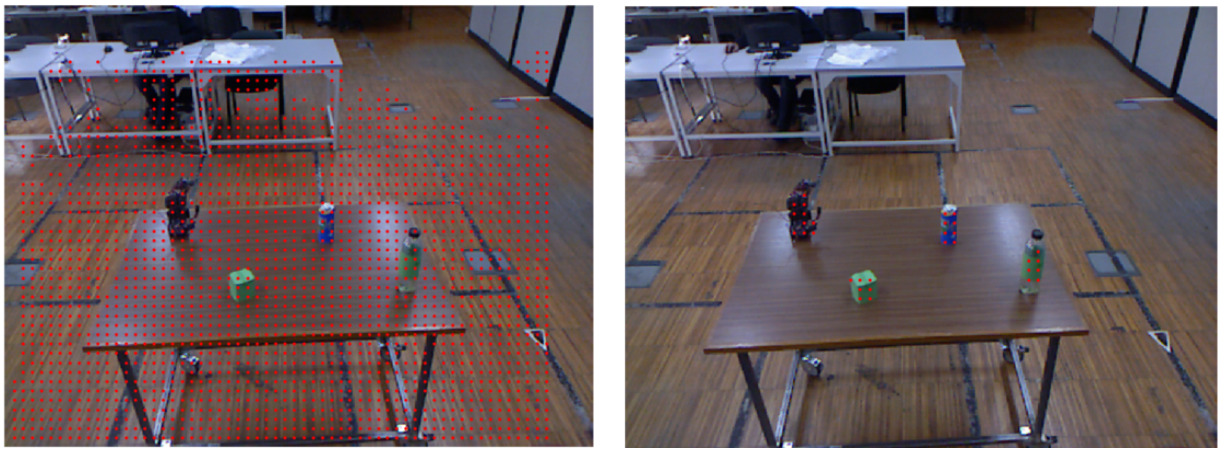
É fácil compreender à partida que com este tipo de abordagem apenas são precisas algumas amostras para gerar estimativas de pose, isto torna o método mais robusto a detectar objectos na imagem que estejam parcialmente obstruídos. A liberdade para cada amostra poder gerar um número variável de votos [0,k] acaba por ser uma vantagem, pois se cada amostra votasse sempre um número fixo de vezes, o espaço de votação seria muito mais ruidoso.

4 Implementação do método

Neste capítulo será feita uma descrição mais detalhada da nossa implementação do método de [Kehl et al., 2016], descrito no capítulo anterior.

4.1 Segmentação

A primeira etapa deste método passa por gerar amostras locais de uma imagem RGB-D, invariantes no tempo e com as camadas de cor e profundidade normalizadas. Para isto é considerada uma grelha de pontos sobreposta na imagem e para cada ponto é tido em conta o seu valor de profundidade.



(a) (b)

Figura 4.1: Centros das amostras considerando (b) ou não (a) o espaço de trabalho.

Visto no nosso caso especifico haver um conhecimento prévio do espaço de trabalho, foi implementado um passo extra para remover pontos da grelha fora desse volume, tendo em conta que a grande maioria dos pontos da imagem pertencem ao ambiente, este passo extra permite-nos poupar significativamente recursos computacionais a analisar amostras irrelevantes. O tamanho de cada amostra é calculado de acordo com a equação 3.1, a normalização é feita de forma a que o intervalo de valores para cada camada de cor seja

$[-1, 1]$ e para a camada de profundidade $[-m, m]$, no método original afirmam que deste modo a extracção de amostras locais evita a necessidade de lidar com problemas de modelação do ambiente.

4.2 Extração de Características e Lista de Referências

Redes Neuronais Convolucionais (CNN)

As redes neuronais convolucionais (CNN), são arquitecturas que estão a ser fortemente investigadas e têm-se tornado ao longo dos anos o estado da arte em várias áreas de investigação, como por exemplo, visão por computador, reconhecimento e interpretação de fala natural, entre outras. Este tipo de arquitectura tira partido da capacidade de aproximarem filtros convolucionais, capazes de extrair características relevantes e combina-las de forma a classificar as suas entradas. É normal utilizar redes com arquitecturas de multi perceptrão, acopladas à componente convolucional, para realizar a classificação da informação, este tipo de abordagem obriga com que as dimensões da informação de entrada sejam fixas, no entanto devido à natureza convolucional, arquitecturas puramente convolucionais não apresentam esta restrição. Tipicamente as CNN's são treinadas de maneira supervisionada, ou seja é necessário a existência dum *dataset* com informação previamente classificada, normalmente por humanos, de modo a conseguir extrapolar características relevantes à classificação.

Autoencoders Convolucionais(CAE)

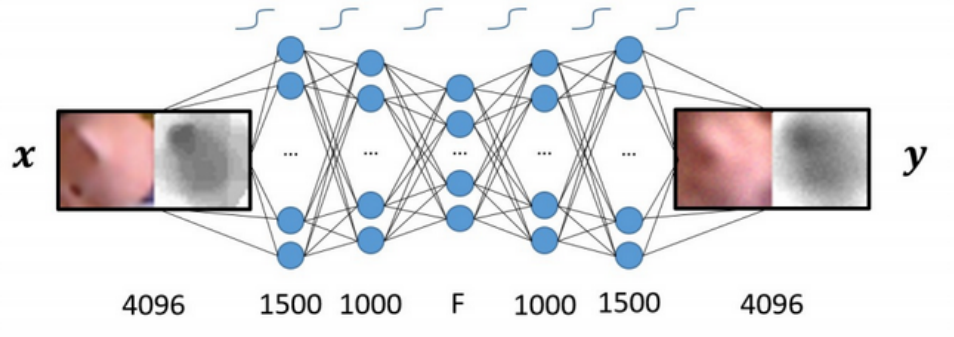


Figura 4.2: Exemplo de um AE, figura retirada de [Kehl et al., 2016]

Uma arquitectura *Autoencoder* (AE) utiliza as suas camadas centrais para aprender uma codificação compacta da informação à sua entrada, tem como principio fundamental mini-

mizar o erro entre a descodificação dessa representação e a informação original que entrou na rede $||x - y||$, de forma a garantir uma codificação relevante da informação. Este tipo de abordagem não é supervisionada, ou seja não é necessário informação previamente classificada, e pode ser aplicada a todo o tipo de tarefas que necessitem de uma representação compacta de informação, como por exemplo classificação de imagens.

No caso dos AE, uma imagem precisa ser desdobrada num vector finito de entradas, o que introduz alguma redundância ao processo e obriga de certa forma a que as características extraídas sejam globais. Quando consideramos uma arquitectura AE convolucional (CAE), devido à natureza das convoluções a rede consegue escalar bem em relação às dimensões de entrada, porque o número de parâmetros para gerar mapas de activação é constante e não introduz redundância no sistema, o que torna as CAE extractores de características generalizadas.

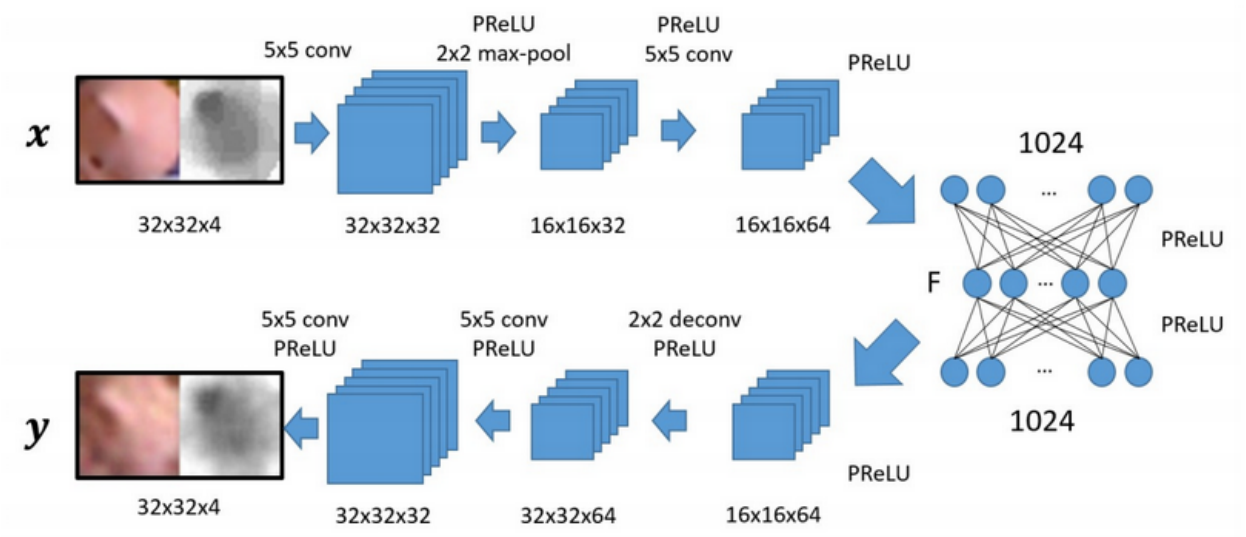


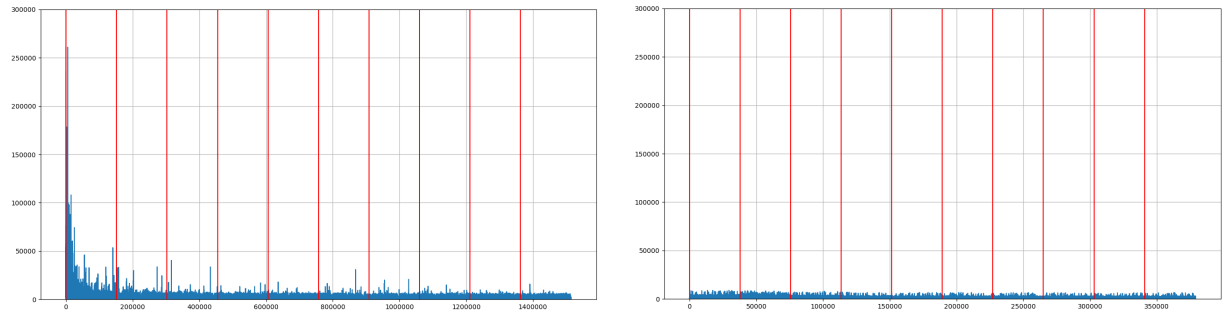
Figura 4.3: *Autoencoder* convolucional (CAE) implementado, figura retirada de [Kehl et al., 2016]

Treino e Validação do Extractor de características CAE

Tal como proposto originalmente, foi implementada a arquitectura descrita na $fig.4.3$, que passa por aplicar, a uma amostra, múltiplos filtros convolucionais 5×5 seguidos de funções rectificadoras ($PReLU$'s), acoplados a uma rede neuronal simples multi camada, com duas partes "simétricas", uma encarregue de codificar os resultados das convoluções num vector de características com dimensão F e a outra de descodifica-lo. A reconstrução da imagem de entrada é obtida aplicando uma desconvolução ao vector de características

descodificado, pela rede multi-camada, com filtros 2×2 seguido de $PReLU$'s e aplicando novamente convoluções 5×5 seguidas de $PReLu$'s. Durante a fase de codificação é aplicado uma operação de *max-pool*, com janelas 2×2, este passo não só reduz significativamente o número de parâmetros envolvidos, como generaliza melhor os resultados das convoluções tornando a detecção de características mais robusta em termos de escala e deslocamentos da entrada.

Para treinar o extractor de características CAE foi utilizado o dataset disponibilizado por [Lai et al., 2011] com múltiplas vistas RGB-D para 51 categorias de objectos, cada uma com múltiplas instâncias, por exemplo para a categoria *lata* existem imagens de múltiplas vistas das instâncias *lata de pepsi*, *lata de Coca-Cola*,... Tirando partido do dataset [Lai et al., 2011] todas as amostras utilizadas são de objectos, devido à fase de normalização das amostras, imagens do ambiente de fundo não devem gerar características relevantes. De reparar ainda que a rede é treinada com amostras de imagens, tendo em conta que o dataset disponibiliza uma média de 200 imagens RGB-D para cada uma das 300 instâncias, dependendo da densidade de segmentação, o número das amostras de treino e validação facilmente anda na ordem dos milhões.



(a) (b)

Figura 4.4: Erro quadrático entre a reconstrução e a entrada da rede por amostra, durante a fase de treino (a) e validação(b). O inicio de cada época é representada com uma linha vertical vermelha.

O treino foi feito de modo clássico, o *Dataset* foi dividido em 2 porções de 80% e 20%, para treino e validação da rede respectivamente. Para avaliar a qualidade da reconstrução foi tido em conta o erro quadrático entre a sua entrada e a sua saída, durante a fase de treino a optimização dos parâmetros internos foi feita tendo em conta o algoritmo de optimização de Adam, descrito em mais detalhe em [Kingma and Ba, 2014]. Cada época passa por uma

etapa inicial de treino onde a rede "vê", numa ordem aleatória, todas as amostras dos 80% do *dataset* dedicadas ao treino e optimiza os seus parâmetros de forma a minimizar o erro da saída. Após serem "vistas"todas as amostras do treino, entramos na fase de validações com as restantes amostras, idênticas mas previamente não vistas. Nesta etapa a rede não optimiza os seus parâmetros, apenas avalia a sua performance e caso seja melhor que a da última época, é guardada a configuração actual e iniciada uma nova época.

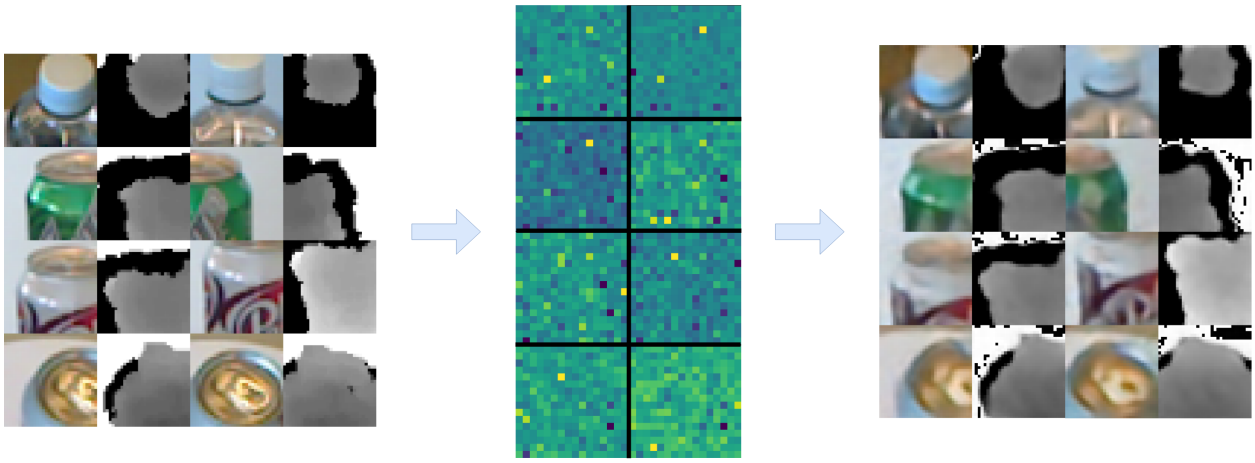


Figura 4.5: Exemplo de amostras forneciadas à rede CAE(à esquerda), respectivas encriptações(no centro, representação 16×16 dos vectores de 256 caracteristicas) e recontruções das entradas(à direita) após o treino da rede.

Arquivo de amostras sintéticas (*Codebook*)

Com o extractor de Características CAE treinado, vamos então gerar uma lista (*codebook*) de características relativas a múltiplas amostras de objectos sintéticos. Cada entrada da lista deve ter um vector de características acopladas a uma transformação de corpo rígido, entre o referencial do objecto amostrado, centrado no seu centro de massa, com orientação e posição relativas ao referencial da camera, na seguinte representação [x, y, z, raw, $pitch$, yaw].

O *codebook* foi gerado a partir de *datasets* que disponibilizam imagens *RGB-D*, de múltiplas vistas, em conjunto com informação da posição e pose de objectos sintéticos. Os datasets disponibilizados por [syn,], encontram-se todos previamente estandardizados o que facilitou a sua integração na nossa implementação. Foram utilizados nomeadamente os *datasets* de Hinterstoisser [Hinterstoisser et al., 2012] com as melhorias do *groundtruth* de Brachmann [Brachmann et al., 2014], de Doumanoglou [Doumanoglou et al., 2016], de Rutgers [Rennie et al., 2016] e de Tejani [Tejani et al., 2014], no total utilizamos em média 2000

imagens RGB-D para cada um dos 56 modelos sintéticos.

Foram gerados 3 *codebooks* com tamanhos de $1GB$, $500Mb$ e $350Mb$, com o intuito de testar a correlação entre o seu tamanho, a qualidade das detecções e o tempo de execução. Todos os *codebooks* contem amostras de todos os objectos sendo a única coisa que os distingue a densidade com que as imagens foram segmentadas, de cada objecto foram extraídas em média cerca de 17.000, 8.500 e 5.500 amostras respectivamente.

4.3 Votação e Filtragem

O processo para estimação de poses passa por uma etapa de votação, onde cada amostra gera de 0 a K votos, seguida duma de filtragem para remover votos ruidosos e incertos, mais uma vez a nossa implementação é idêntica à original.

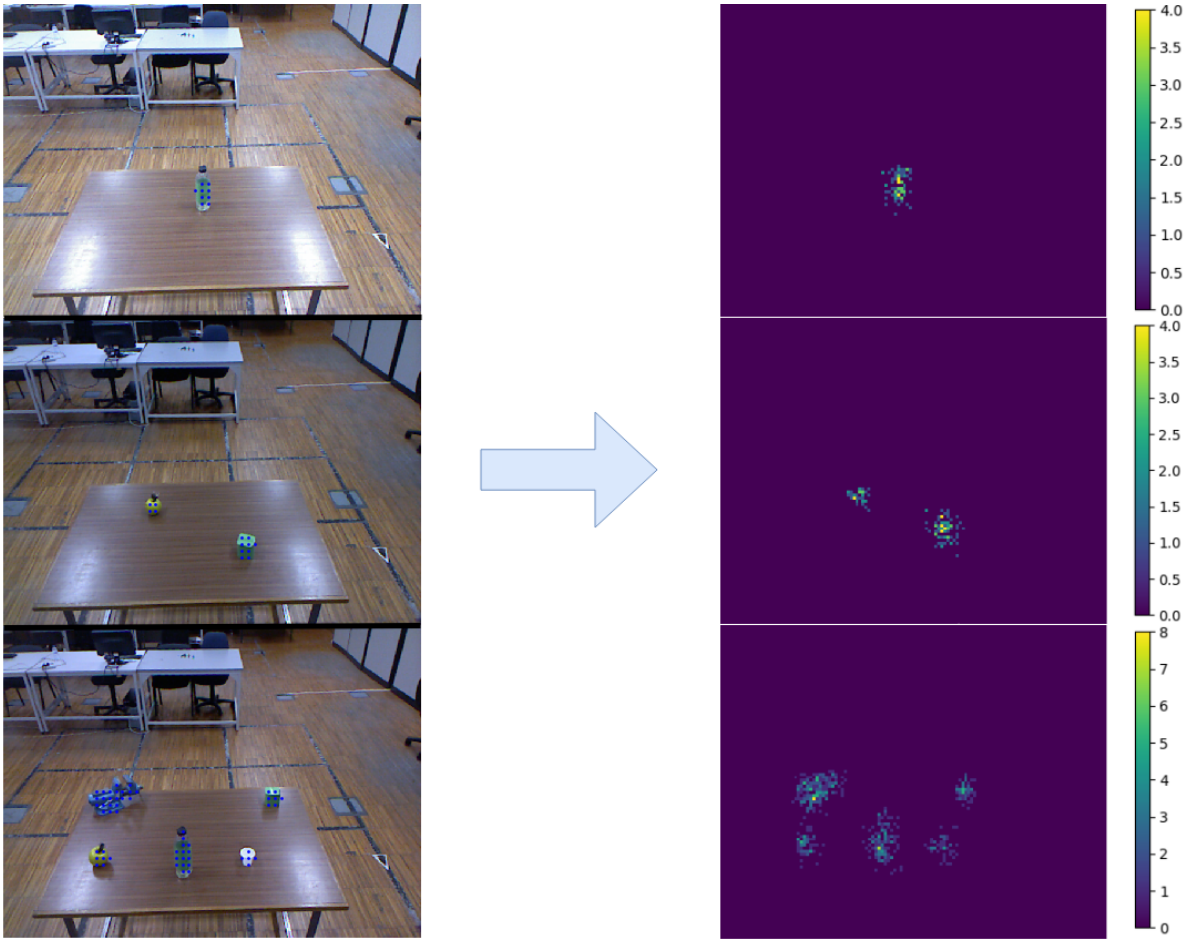


Figura 4.6: Espaço de trabalho (esquerda). Espaço de votos(direita)

Inicialmente é considerada uma grelha de células, equivalente ao espaço de votação, onde cada célula equivale a uma janela de 5×5 pixeis da imagem original. É feita uma pesquisa

no *codebook* dos exemplares mais semelhantes às amostras actuais. Esta pesquisa é feita utilizando um algoritmo dos "K Vizinhos mais próximos"(KNN), onde para cada amostra real, são retornadas as K amostras do *codebook* com a menor distância euclidiana, no espaço de características. É preciso ter em atenção que para cada amostra é necessário pesquisar o *codebook*, isto pode ser indesejado pois implica que o tempo de execução está directamente correlacionado à eficiência da pesquisa e ao tamanho do *codebook*. Apesar de cada amostra, inicialmente com $32 \times 32 \times 4 = 4096$ valores, ser compactada num vector de características significativamente mais pequeno, com 256 valores mais 6 graus de liberdade, tendo em conta que o número de amostras anda na ordem dos milhões, é de esperar que esta etapa consuma grande parte do tempo de execução.

Após a pesquisa K-NN, todos os candidatos gerados por uma amostra com distância euclidiana maior que um dado limiar são descartados. Para os restantes, a sua componente translacional é somada ao vector 3D relativo ao píxel central da amostra, o ponto resultante é reprojectado para imagem e acumulado nas células correspondentes, como pode ser visto na figura 4.6. Com intuito a melhorar os tempos de pesquisa, utilizamos um algoritmo dos K-NN aproximados, disponibilizado no módulo $FLANN$[Muja and Lowe, 2009].

Cada voto tem uma confiança exponencialmente proporcional à sua distância euclidiana da amostra real, no espaço de características, da forma $e^{-||f(x)-f(y_k)||}$, onde $f(x)$ representa as características extraídas da amostra x e $f(y_k)$ os K vectores extraídos do *codebook*. A confiança de cada célula do espaço de votação é simplesmente a soma das confianças de todos os votos que nela se encontram.

Convertido o espaço de votação, de contador de votos para acumulador de confianças, é feita uma filtragem dos votos de modo a descartar células pouco relevantes. Esta filtragem passa por aplicar uma interpolação bilinear $2D$ ao espaço de confianças, seguido de uma detecção dos máximos locais, do novo espaço interpolado (representados na figura 4.7).

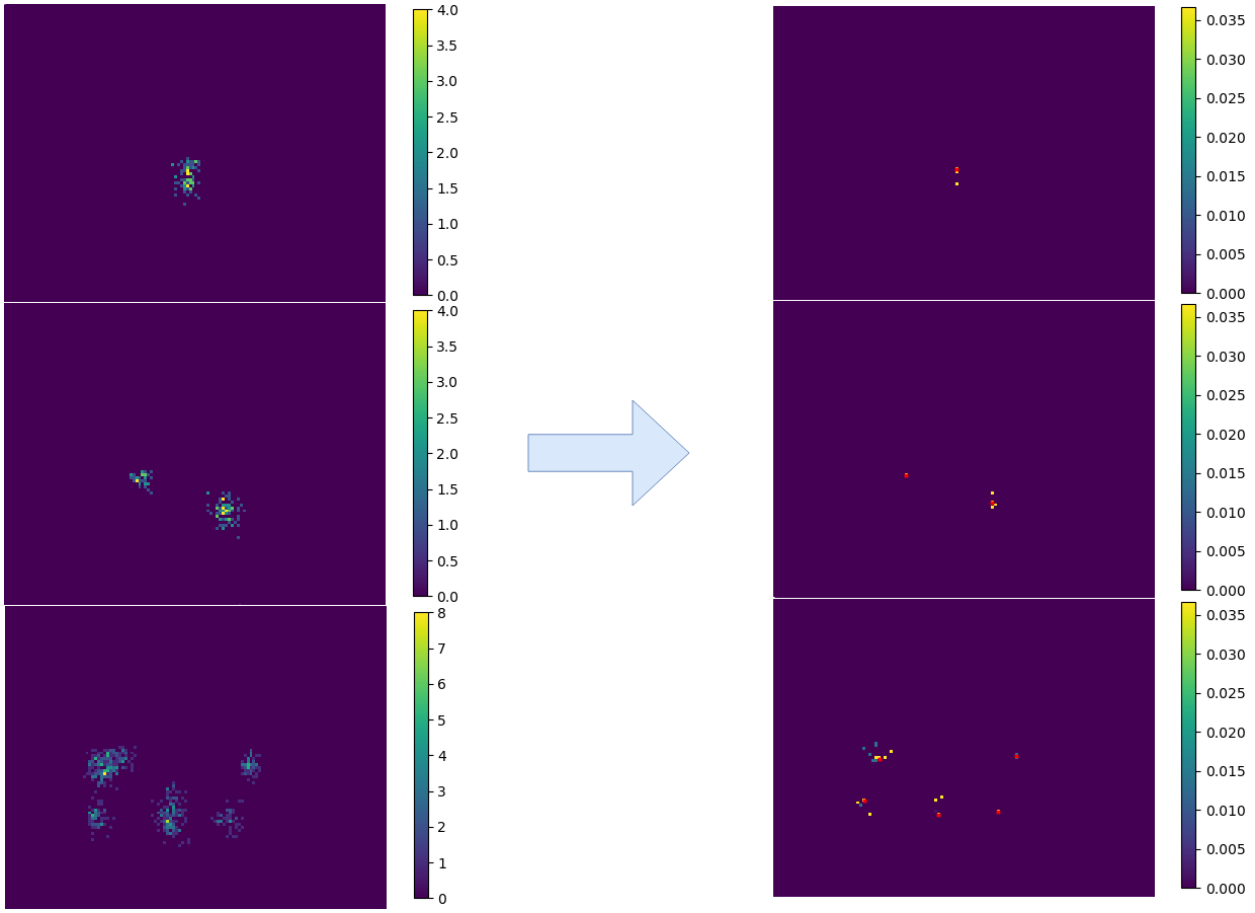


Figura 4.7: Contador de votos (esquerda). Respectivas confianças acumuladas(direita) e máximos locais marcados a vermelho.

4.4 Resultados e Observações

Tendo em conta a dificuldade em arranjar imagens, previamente classificadas, de objectos reais com a sua informação de pose, tornou-se difícil avaliarmos com precisão os resultados do método implementado. Foram utilizadas imagens da *kinect* e tendo em conta os pontos $3D$ dos centros das amostras extraídas, avaliamos os resultados obtidos de forma empírica.

Foi possível extrapolarmos algumas conclusões relativamente ao tempo de execução do método e à relação entre a qualidade aparente dos resultados obtidos. Uma vez que os resultados não são conclusivos, é necessária a realização de testes mais rigorosos.

Em relação às predições obtidas os resultados são ligeiramente inconsistentes. Para objectos reais com características semelhantes aos modelos sintéticos utilizados para a geração dos *codebooks*, as predições aparentam ser coerentes com o esperado. A componente de orientação estimada apresenta algumas variações e inconsistências, enquanto que, os centros de massa aparentam apenas um pequeno erro na ordem dos milímetros.

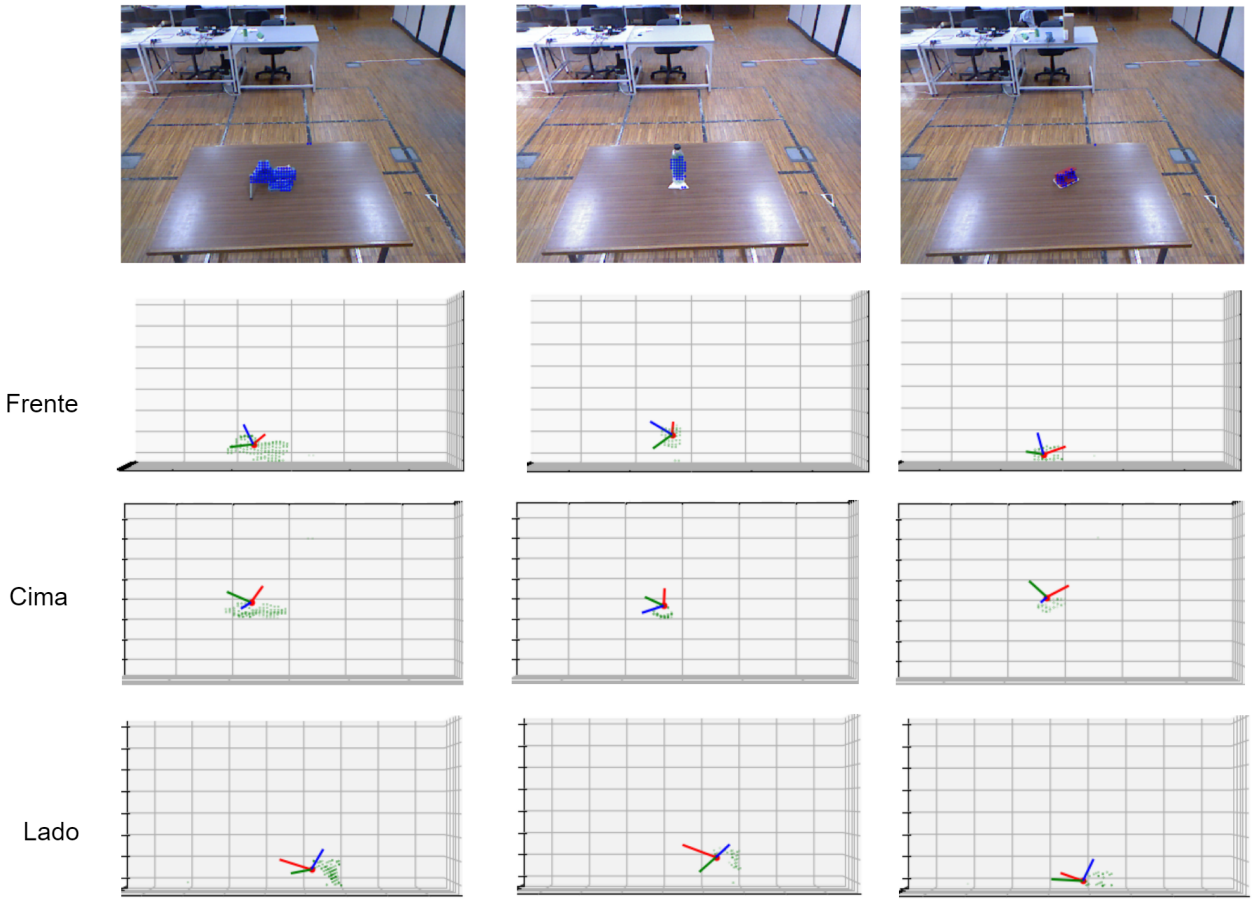


Figura 4.8: Algumas estimativas obtidas pela nossa implementação de [Kehl et al., 2016].

É preciso ter em conta que, apesar de alguns objectos utilizados nos testes terem muitas parecenças com os objectos sintéticos do *codebook* nenhum é igual. Isto pode contribuir para a grande instabilidade da componente de orientação visto não haver nenhum mecanismo que defina o sistema de coordenadas de objectos previamente desconhecidos.

Para objectos desconhecidos ou muito diferentes dos sintéticos as predições pioram significativamente, podendo por vezes as estimativas de pose geradas apresentarem variações muito bruscas, não serem sequer geradas predições ou então serem geradas múltiplas poses para o mesmo objecto.

Estes tipos de inconsistências tanto no número de estimativas como na componente de orientação, podem estar relacionados a múltiplos factores intrínsecos ao método, e também com a forma como certas etapas foram implementadas.

Como já foi previamente referido num capítulo anterior, foi utilizado um algoritmo aproximado para a pesquisa dos candidatos, após uma análise dos resultados desta etapa deparamos que este algoritmo, disponibilizado pela livraria $FLANN$ [Muja and Lowe, 2009], tira

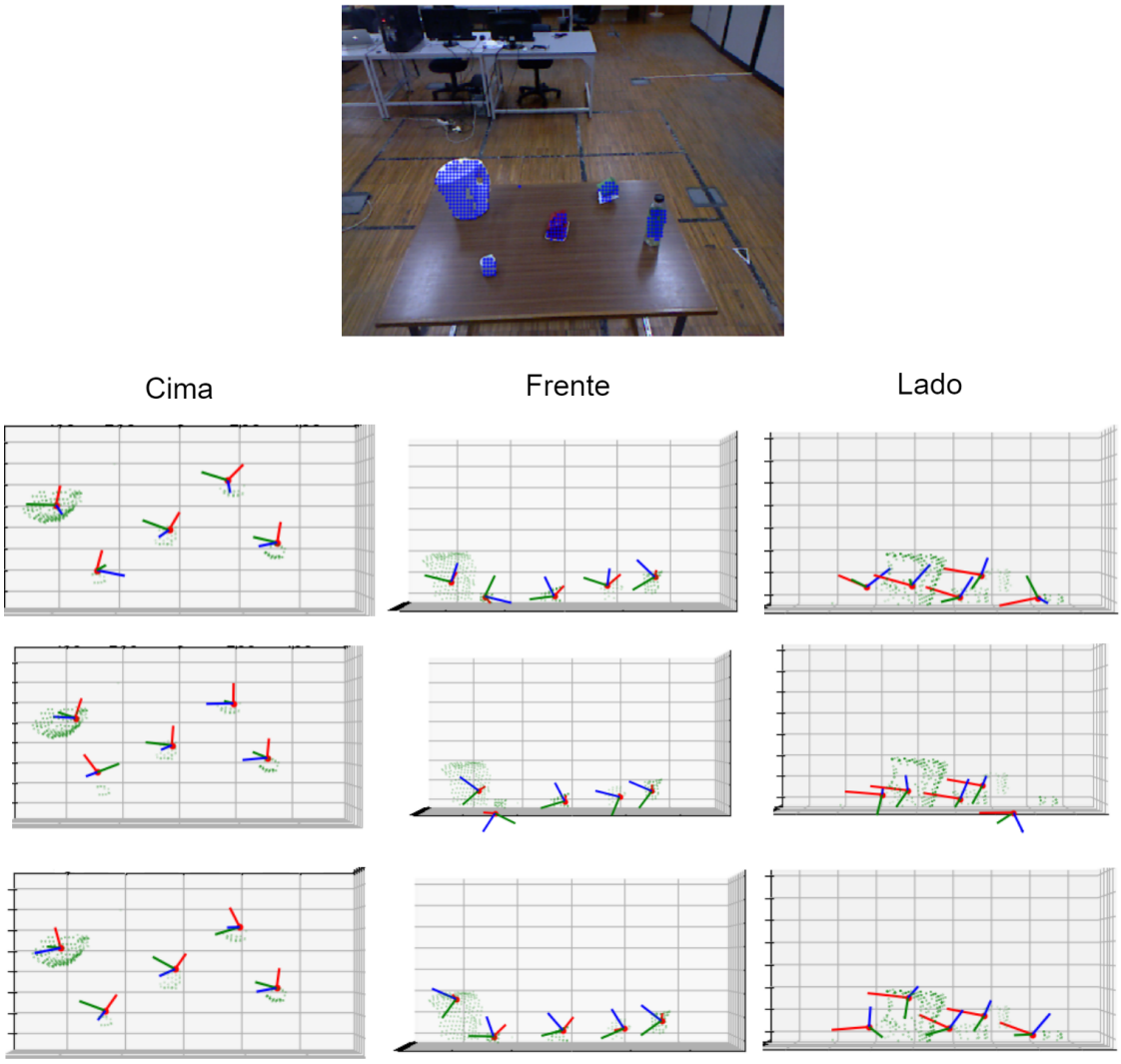


Figura 4.9: Exemplo duma cena analisada 3 vezes consecutivas, onde é possível observar estimativas com posições carteseanas aparentemente consistentes e orientações com variações bruscas.

partido de factores aleatórios tendo assim como consequência directa a geração de candidatos diferentes para condições iniciais iguais. Acreditamos que esta etapa está directamente relacionada às grandes variações nas orientações estimadas, não sendo a única causa mas a que mais contribui para este tipo de inconsistência.

Pensamos ainda que as etapas de segmentação da imagem, atribuição e filtragem das confianças para cada candidato, são responsáveis por não serem geradas predições ou serem geradas predições a mais. O facto da densidade de segmentação ser constante, resulta em objectos equivalentes a uma grande área na imagem (mais perto), gerem significativamente mais amostras do que objectos mais pequenos (mais longe). Estas amostras apesar de serem partes do mesmo objecto, podem não apresentar características semelhantes entre si, o que pode resultar em votos idênticos por amostra mas não por objecto, gerando assim picos no

espaço de confianças. A este espaço é posteriormente aplicada uma interpolação bilinear, e células com confianças suficientemente inferiores a estes picos, podem ser ignoradas não gerando assim predições, ou em contra partida, uma grande área do espaço de confianças correspondente a um só objecto, pode ficar com vários picos resultando assim em múltiplas predições.

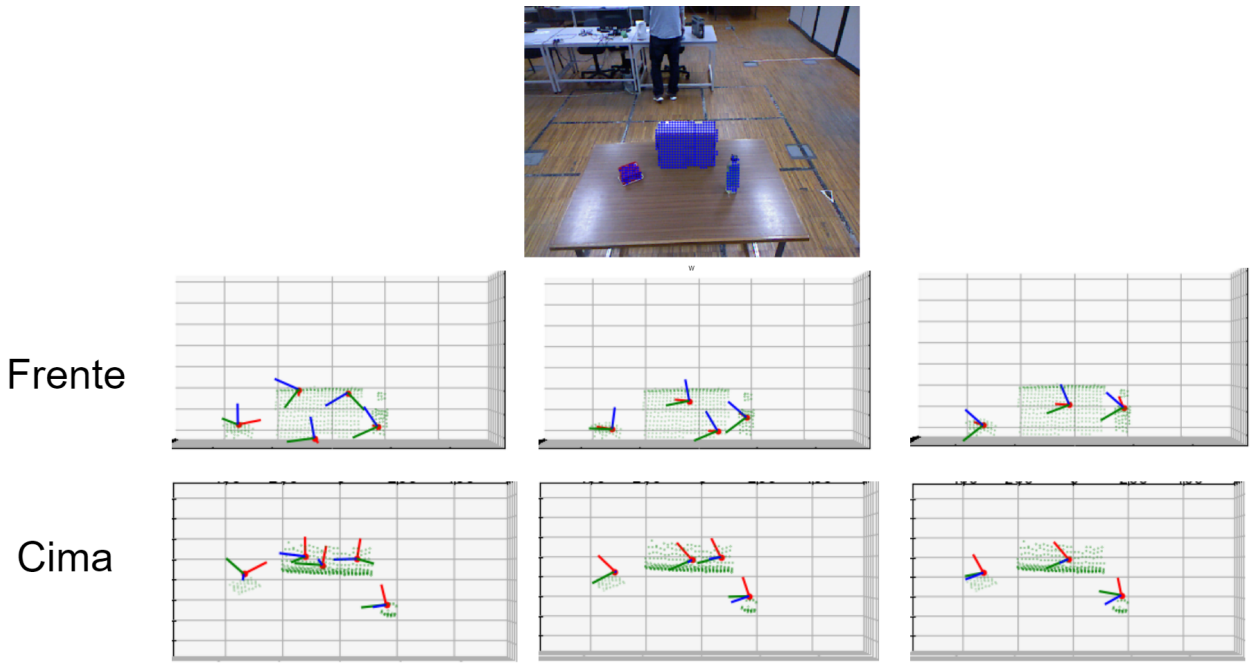


Figura 4.10: Exemplo duma cena analisada 3 vezes consecutivas, onde é possível observar inconsistências no número de estimativas obtidas.

A etapa de pesquisa do *codebook* é sem dúvida a etapa mais crítica da nossa implementação em relação ao tempo de execução. Como já era previamente esperado é a que consome a maior fatia temporal e é muito dependente do tamanho do *codebook* utilizado. Ao utilizarmos *codebooks* com uma quantidade maior de amostras, tanto o número de votos por amostra como o tempo de execução tende a aumentar significativamente, no entanto isto não implica uma melhoria clara nas estimativas de pose obtidas.

A etapa de extracção de características também requer tempo substancial, no entanto isto deve-se maioritariamente às limitações do hardware utilizado na realização dos testes. A rede CAE foi treinada numa placa gráfica NVIDIA 1080 gtx, resultando em tempos de extracção das características cerca de três ordens de grandeza inferiores. Tendo em conta que no nosso *setup* a rede está a ser processada no *cpu* em vez de na *gpu*, este aumento temporal está de acordo com o que é esperado.

5 Modificações ao método original

Apesar dos testes realizados serem fundamentalmente empíricos, não temos qualquer dúvida acerca da necessidade de substituir a pesquisa $K\text{-}NN$. Como foi referido no capítulo anterior esta etapa da implementação original, não só consome uma fatia temporal significativamente maior do que qualquer outra etapa, como pensamos que as suas propriedades aleatórias estão fortemente relacionadas com a instabilidade dos resultados.

5.1 Preditores

Propomos assim, como alternativa a pesquisar os *codebooks* para possíveis candidatos, utilizar os mesmos *codebooks* como *dataset* para o treino de redes neuronais (preditores). A ideia passa por ver se as redes são capazes de extrapolar uma relação entre os vectores de características e as estimativas de pose a si associadas, e dado um novo vector, serem capazes de fazer uma estimativa da pose do objecto correspondente à amostra local.

Tendo em conta que o tempo de execução dum algoritmo de pesquisa é fundamentalmente dependente da quantidade de informação a pesquisar, e que o tempo de propagação duma rede neuronal é apenas dependente do seu número de parâmetros intrínsecos, são esperadas melhorias significativas em relação ao tempo de execução global. É ainda esperado o aumento da concordância das predições entre $frames$ consecutivos, uma vez que esta abordagem não apresenta qualquer tipo de factor aleatório.

Arquitecturas propostas

Nesta fase da dissertação testamos algumas arquitecturas e fizemos uma breve análise do seu erro de treino e das detecções geradas, de forma a tentar perceber se as redes conseguem extrapolar uma relação entre características locais e poses de objectos.

Foram treinados 4 preditores simples com arquitectura de perceptrão multi camada, onde para cada foi acrescentada uma camada intermédia entre a ultima e penúltima camada tal

como descrito na figura 5.1.

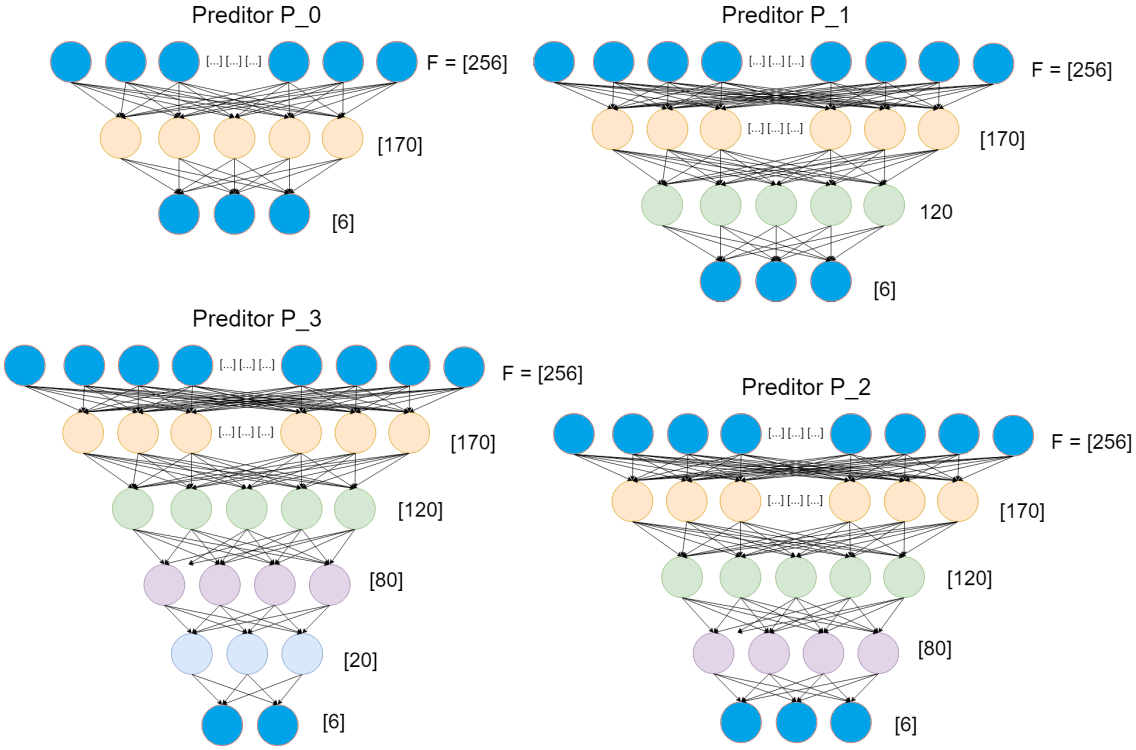


Figura 5.1: Arquitecturas perceptrão multicamada utilizidas.

Treino e Integração no método

Os preditores foram treinados tirando partido dos múltiplos *codebooks* previamente obtidos como *dataset*. Para cada preditor foram feitas 4 épocas de treino onde em cada consideramos 80% das amostras de cada *codebook*, escolhidas de forma aleatória. Os parâmetros da rede foram optimizados tendo em conta o gradiente estocástico do erro quadrático das estimações feitas.

Nenhuma das arquitecturas propostas mostrou ter a capacidade de extrapolar uma relação entre as características e as suas poses correspondentes e o facto de utilizarmos um número de amostras na ordem dos milhões, de múltiplos objectos sem relação aparente entre si contribui para este problema.

Durante o treino é possível ver que o erro por iteração, apresenta oscilações bruscas e nunca tende a diminuir, no entanto, uma análise do erro relativamente às amostras em si aparenta que as redes têm uma maior capacidade para extrapolar uma relação pertinente para certos objectos mais do que outros.

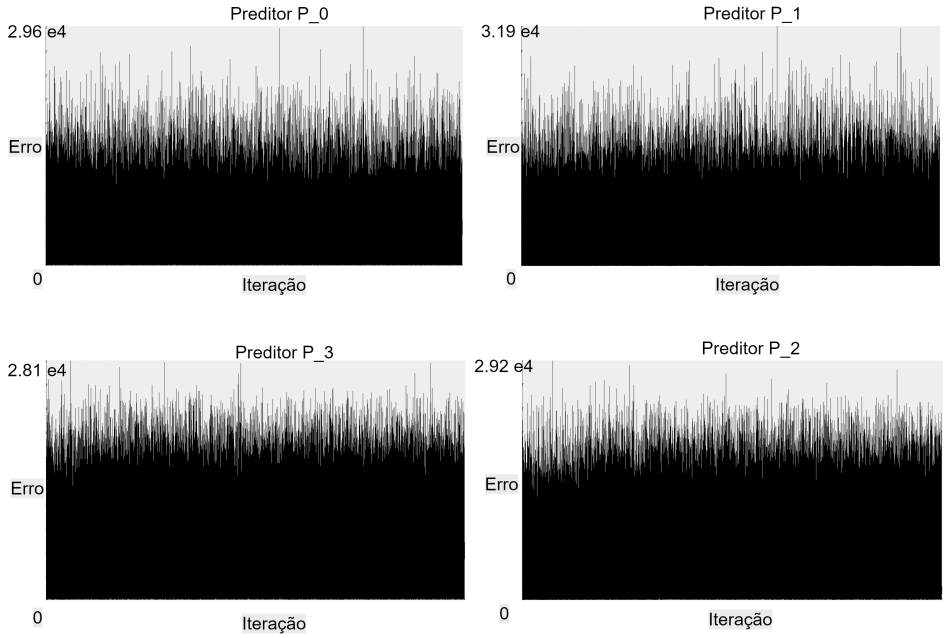


Figura 5.2: Erro do treino dos preditores por iteração

Independentemente dos maus resultados durante a etapa de treino, foi implementada uma variação da implementação inicial, onde integramos as redes obtidas no lugar da pesquisa $K\text{-}NN$, com o intuito de comparar o tempo de execução do algoritmo em relação à implementação original e para que numa futura continuação desta dissertação, novas arquitecturas possam ser exploradas e facilmente integradas no algoritmo.

Novo espaço de votos

As redes utilizadas permitem gerar estimativas para a posição relativa a cada amostra de forma mais rápida do que na implementação inicial, em contra partida, a pesquisa $K\text{-}NN$ não só gera múltiplos candidatos por amostra, como uma estimativa da credibilidade de cada um através das suas distancias euclidianas no espaço de características. Visto as redes preditoras apenas gerarem um candidato por amostra, surge a necessidade de alterarmos a forma como o espaço de confianças é calculado.

Assumindo que objectos na imagem simplesmente não aparecem e/ou desaparecem, e que o método possa ser rápido o suficiente para capturar movimentos ao longo de múltiplos $frames$, podemos reutilizar espaços de confiança previamente calculados para colmatar a quantidade inferior de votos.

$$Cell_{with}Centroid = \frac{Number_{of}Votes_{in}Cell}{TotalNumber_{of}Votes} \tag{5.1}$$

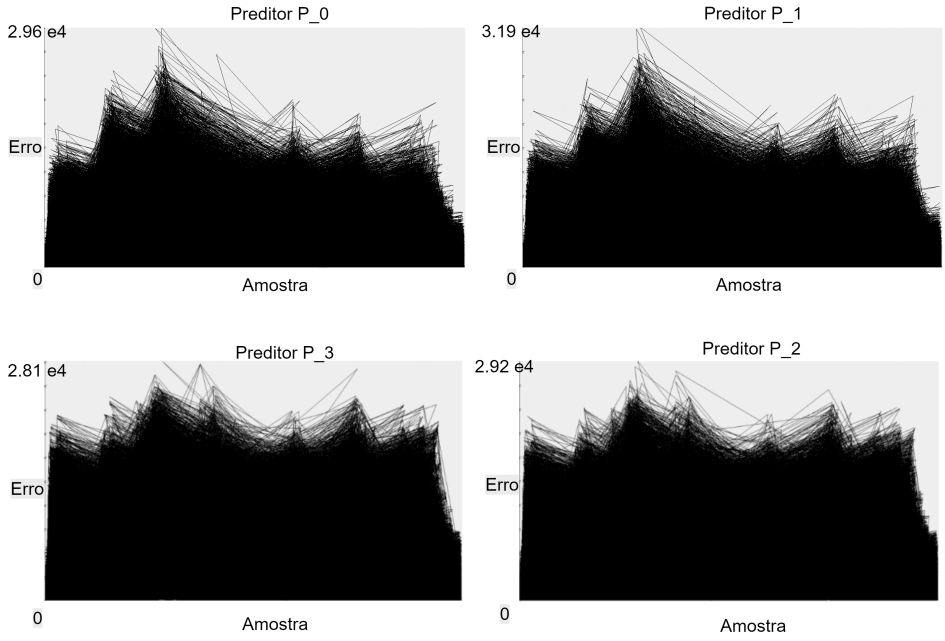


Figura 5.3: Erro do treino dos preditores por amostra do codebook

A nossa ideia passa por calcular para cada *frame* um espaço de credibilidades intermédio de forma semelhante, projectando os votos para uma grelha de células e de seguida atribuindo uma credibilidade a cada célula de conter centros de massa de objectos. Esta credibilidade simplesmente o *ratio* entre o número de votos da célula e do número total de votos do espaço, tal como descrito na 5.1. Resultando assim num espaço de credibilidades baseado no número de votos, onde uma célula que contenha todos os votos tem uma credibilidade de 1, uma que não contenha nenhum voto tenha uma credibilidade de 0, e no caso mais comum, se os votos estiverem distribuídos pelo espaço, as células têm valores dentro do intervalo]0,1[de forma a que somatório do espaço de credibilidades seja 1.

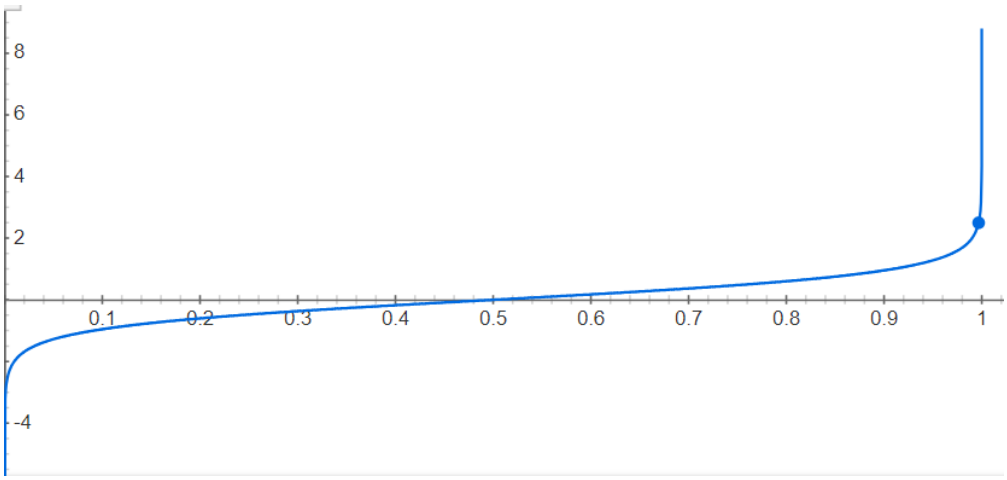


Figura 5.4: Função *Log Odds*

De seguida este espaço é normalizado de forma a que células com votos tenham uma

credibilidade no intervalo]0.5,1[e células sem votos de 0, como descrito pela equação 5.2. Desta forma um conjunto de regras difusas do género,

em células com valores de credibilidade de conter centros de massa próximos:

- de 0, é altamente improvável haver objectos
- de 0.5, é incerto haver objectos
- de 1, é altamente provável existirem objectos

pode ser implementado tirando partido de log odds como difusor (equação 5.3 e figura 5.4). Convertendo finalmente as credibilidades em incrementos de confiança, é actualizado o espaço de confianças global tal como descrito na equação 5.4.

$$CellTrust = \frac{(1 + CellTrust)}{2} \tag{5.2}$$

$$TrustIncrement = \log \frac{CellTrust}{1 - CellTrust} \tag{5.3}$$

$$FinalTrustSpace[k] = FinalCellSpace[k-1] + TrustIncrement \tag{5.4}$$

Por fim as predições são obtidas interpolando o espaço de confianças global e aplicando uma operação de $meanshift$ aos votos dentro das células correspondentes aos máximos locais do espaço interpolado, tal como na implementação inicial.

5.2 Resultados e Observações

O nosso intuito principal era perceber se este novo método proposto conseguia apresentar melhorias significativas no que diz respeito aos tempos de execução, uma vez o mau desempenho obtido na fase de treinos dos preditores. Os testes realizados foram mais uma vez apenas empíricos, pelos mesmos motivos referidos anteriormente, havendo assim a necessidade de serem realizados testes mais rigorosos. No entanto, o método apresentou estimativas aparentemente semelhantes às obtidos pela nossa implementação inicial.

Tempos médios de execução

Relativamente aos tempos de execução médios, as duas implementações foram testadas 10 vezes recorrendo a um conjunto de 45 imagens, cada uma contendo um ou múltiplos objectos, com e sem oclusões. Os valores obtidos são muito dependentes da etapa de segmentação, uma vez que quanto mais segmentada for a imagem, mais amostras são processadas.

A nossa primeira implementação apresentou um tempo médio de execução por imagem de aproximadamente 5.54 segundos, onde as etapas mais criticas são a de pesquisa K-NN, com uma duração média de 2.7 segundos, seguida da etapa de extracção das características, com uma duração média de cerca de 1.55 segundos.

O novo método apresentou um tempo médio de execução por imagem de aproximadamente 1.73 segundos, significativamente mais rápido, onde a etapa mais critica foi a de extracção de características. Os preditores demoram em média cerca de 0.005 segundos a gerarem votos.

Há assim uma melhoria clara nos tempos de execução, em utilizar redes neuronais como alternativa ao algoritmos de pesquisa K-NN.

Predições

Tendo em conta que durante a fase de treino nenhuma das redes preditoras demonstrou ter a capacidade de extrapolar uma relação relevante entre as amostras do espaço de características e as suas respectivas estimativas de pose (5.2), e ainda que, estas redes preditoras são fundamentais para o método proposto, não eram esperadas estimativas relevantes. No entanto os resultados obtidos são de certa forma semelhantes aos da nossa implementação

inicial.

As estimativas obtidas aparentam ser bastante coerentes no que diz respeito à sua componente de posição, e não apresentam variações tão bruscas relativamente à componente de orientação, como pode ser observado na figura 5.5.

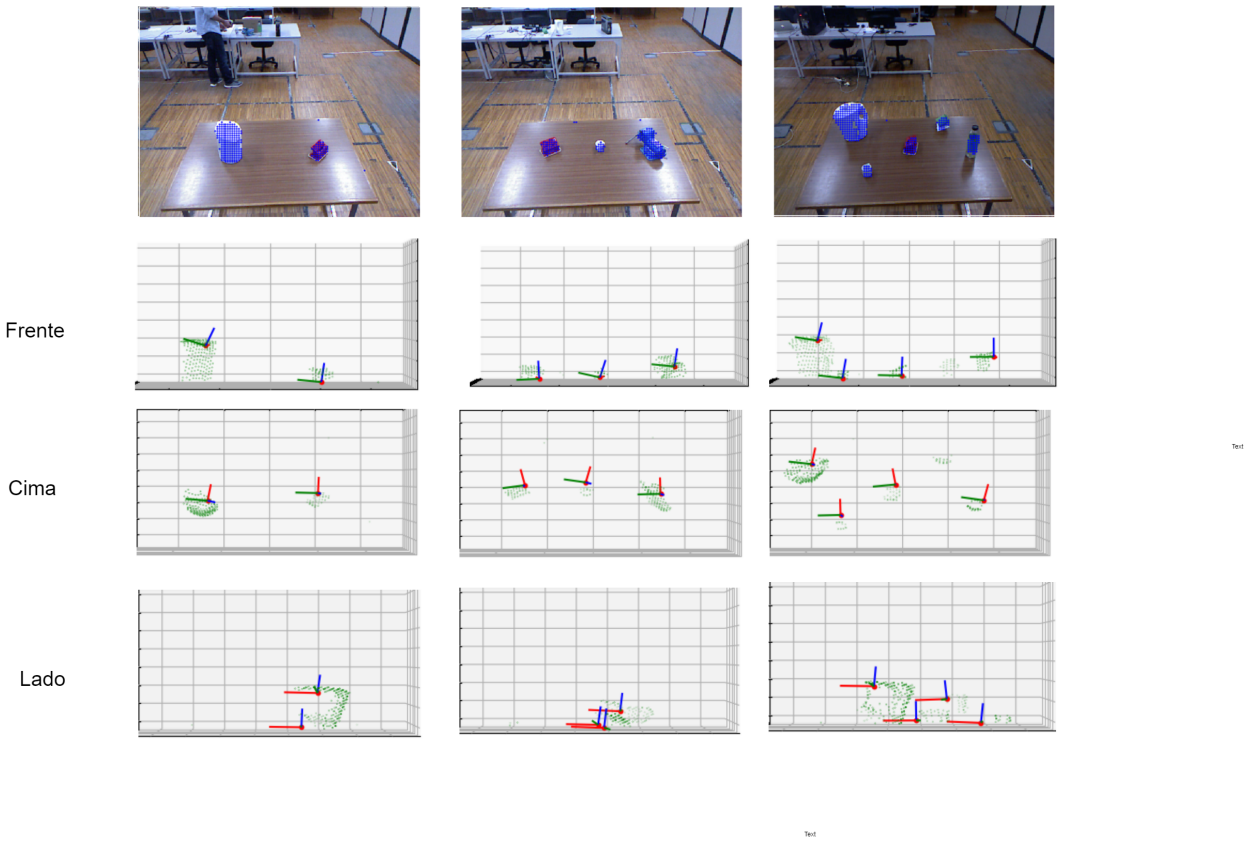


Figura 5.5: Algumas estimativas feitas pelo novo método, tirando d partido de redes neuronais preditoras e do novo espaço de confianças. As estimativas obtidas baseiam-se num espaço de confianças calculado ao longo de 5 frames consecutivos.

6 Sugestões para trabalhos futuros

Sentimos que no futuro é absolutamente necessário obter um *dataset* de imagens *RGB-D* de objectos reais previamente classificados com informação de pose, de forma a ser possível efectuar uma análise mais rigorosa dos métodos implementados.

Em relação aos métodos implementados, seria interessante uma abordagem semelhante que considerasse um espaço de votação $3D$, uma vez que os votos são projectados para o plano de imagem, é possível que objectos alinhados gerem votos para a mesma célula.

No futuro achamos necessário tentar encontrar uma rede que apresente uma melhor capacidade de extrapolar uma relação entre os vectores de características e as poses associadas. Possivelmente alterando o numero de camadas e neurónios por camada, ou as funções de activação dos neurónios, ou ainda arquitecturas diferentes como redes CNN ou RNN.

Visto ainda ter sido a eliminada a etapa de pesquisa, pode já não fazer sentido a necessidade de ter uma representação tão compacta das amostras extraídas. No futuro seria também interessante utilizar representações com um pouco mais de informação e verificar se as redes conseguem extrapolar mais facilmente uma relação entre características e poses.

7 Bibliografia

[syn,] http://cmp.felk.cvut.cz/sixd/challenge2017/. Accessed: 2017-10-30.

[Brachmann et al., 2014] Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., and Rother, C. (2014). Learning 6d object pose estimation using 3d object coordinates. In *European conference on computer vision*, pages 536–551. Springer.

[Doumanoglou et al., 2016] Doumanoglou, A., Kouskouridas, R., Malassiotis, S., and Kim, T.-K. (2016). Recovering 6d object pose and predicting next-best-view in the crowd. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3583–3592.

[Garon and Lalonde, 2017] Garon, M. and Lalonde, J.-F. (2017). Deep 6-dof tracking. *IEEE transactions on visualization and computer graphics*, 23(11):2410–2418.

[Girshick, 2015] Girshick, R. (2015). Fast r-cnn. *arXiv preprint arXiv:1504.08083*.

[Hinterstoisser et al., 2012] Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., and Navab, N. (2012). Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, pages 548–562. Springer.

[Iventosch et al., 2017] Iventosch, J. W. et al. (2017). *A deep learning framework for model-free 6 degree of freedom object tracking*. PhD thesis.

[Kehl et al., 2016] Kehl, W., Milletari, F., Tombari, F., Ilic, S., and Navab, N. (2016). Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation. In *European Conference on Computer Vision*, pages 205–220. Springer.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Lai et al., 2011] Lai, K., Bo, L., Ren, X., and Fox, D. (2011). A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824. IEEE.

[Luo et al., 2014] Luo, W., Zhao, X., and Kim, T.-K. (2014). Multiple object tracking: A review. *arXiv preprint arXiv:1409.7618*, 1.

[Muja and Lowe, 2009] Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pages 331–340. INSTICC Press.

[Pauwels and Kragic, 2015] Pauwels, K. and Kragic, D. (2015). Simtrack: A simulation-based framework for scalable real-time object pose detection and tracking. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 1300–1307. IEEE.

[Redmon et al., 2016] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

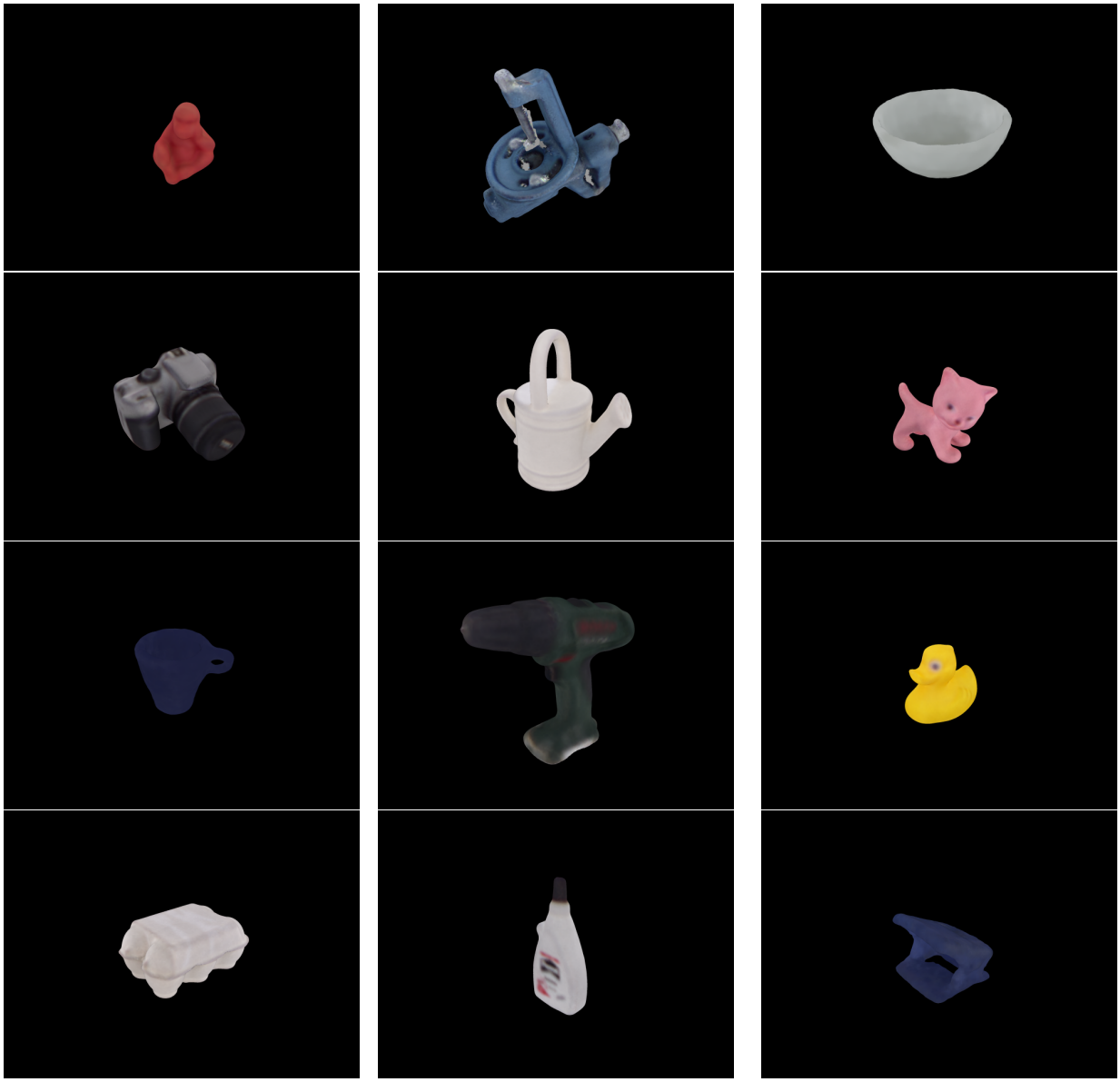
[Rennie et al., 2016] Rennie, C., Shome, R., Bekris, K. E., and De Souza, A. F. (2016). A dataset for improved rgbd-based object detection and pose estimation for warehouse pick-and-place. *IEEE Robotics and Automation Letters*, 1(2):1179–1185.

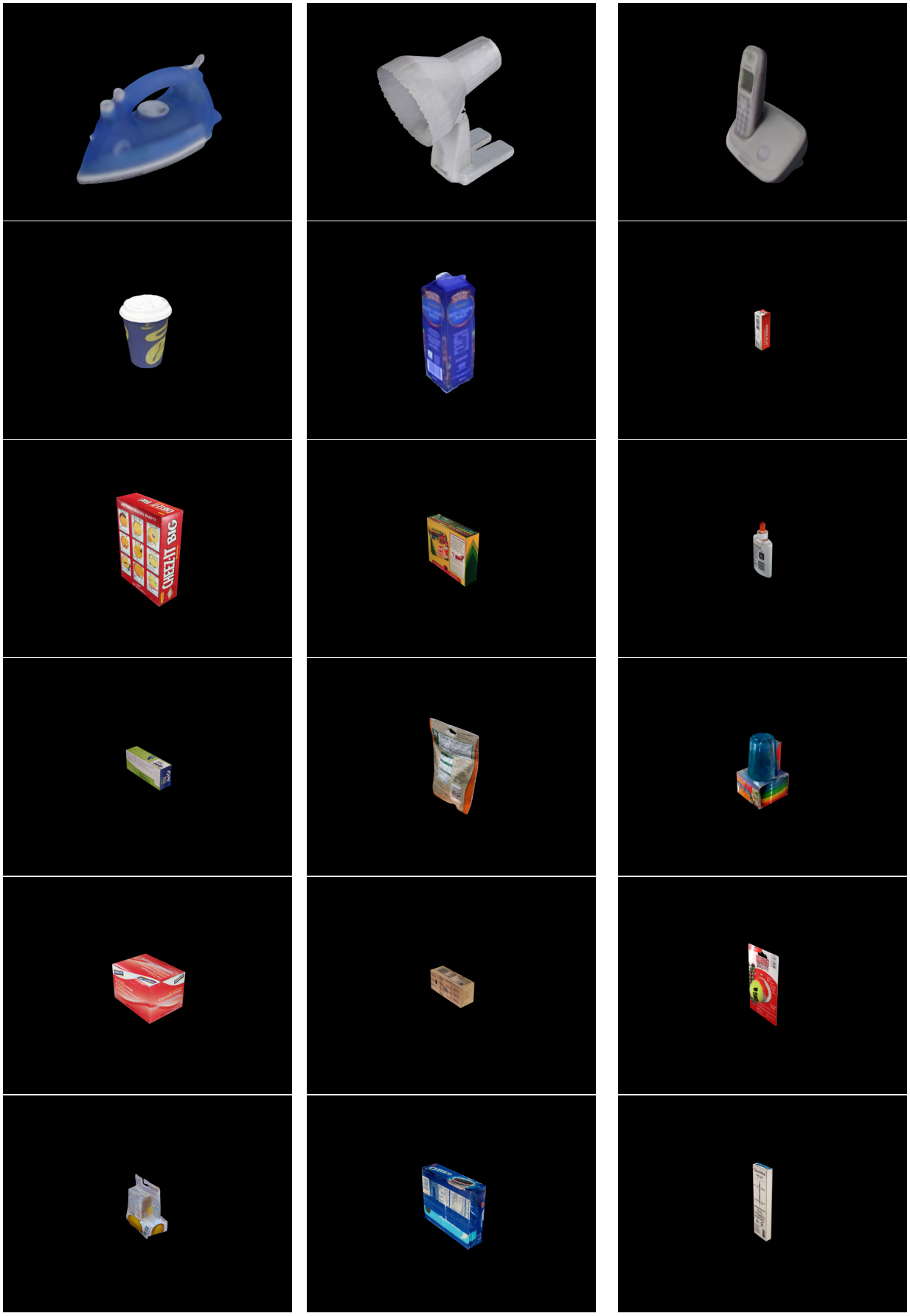
[Tejani et al., 2014] Tejani, A., Tang, D., Kouskouridas, R., and Kim, T.-K. (2014). Latent-class hough forests for 3d object detection and pose estimation. In *European Conference on Computer Vision*, pages 462–477. Springer.

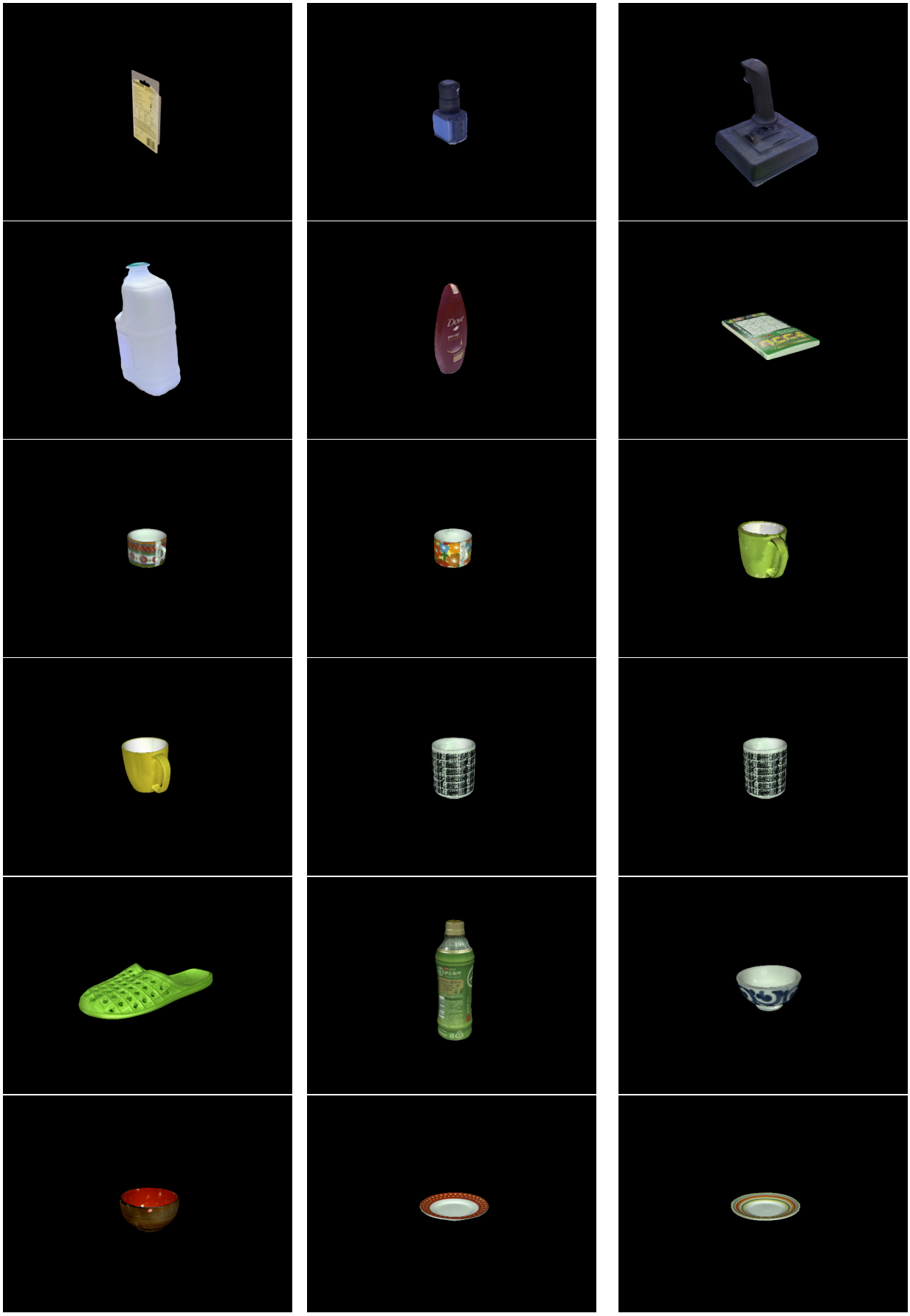
[Yilmaz et al., 2006] Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13.

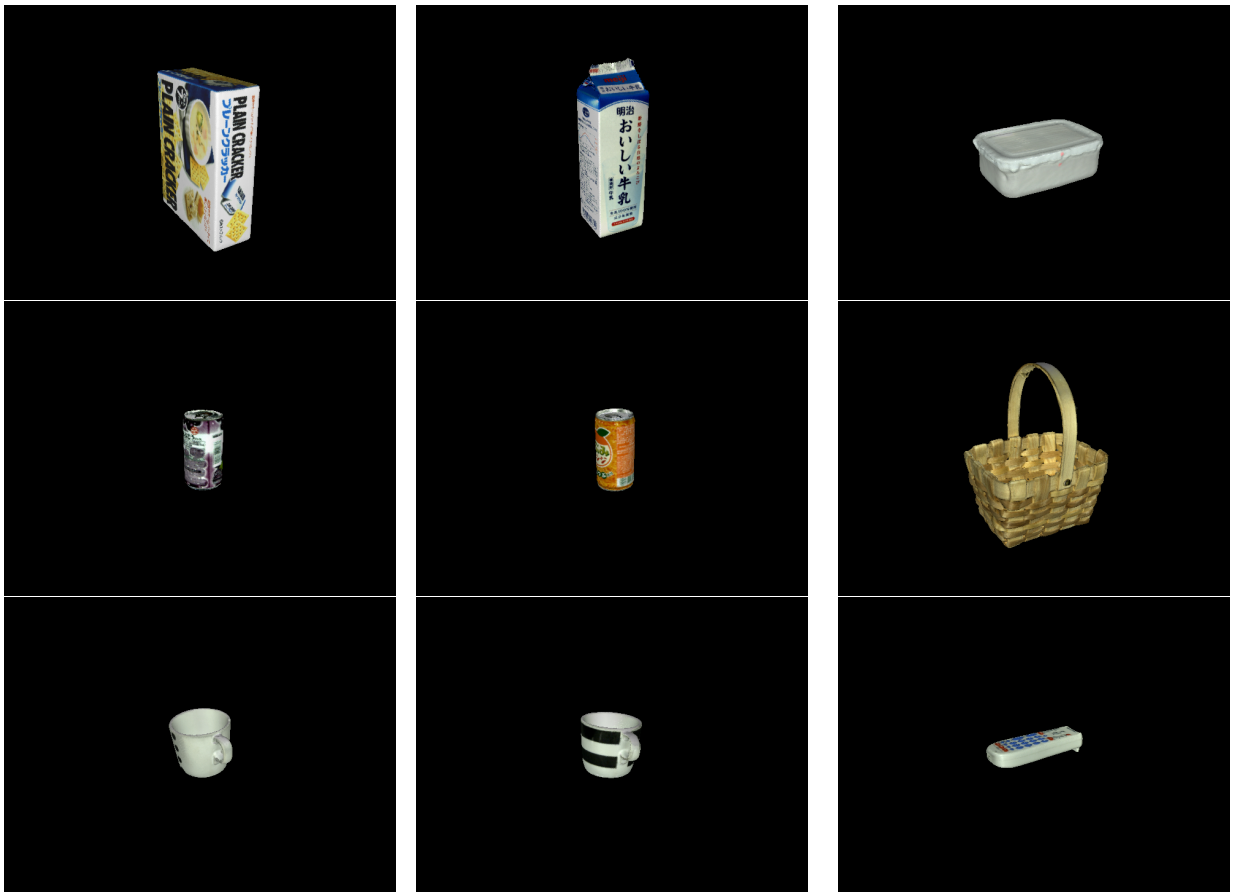
Apêndice

Modelos sintéticos utilizados:




CHEEZ-IT BIG




PLAIN CRACKER
明治
おいしい牛乳