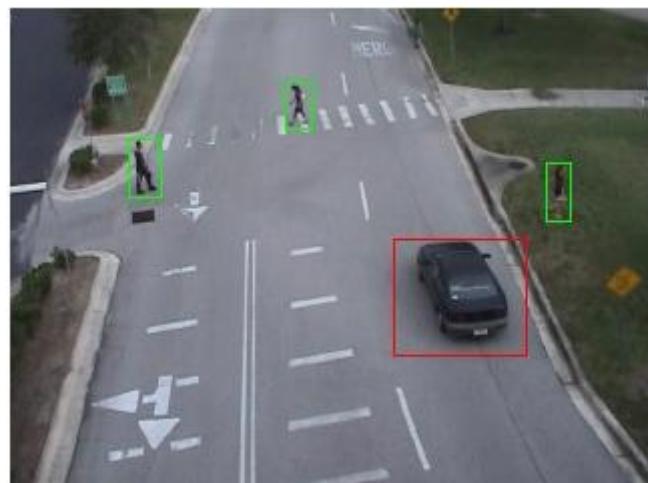


PROCESSAMENTO DE IMAGEM E VISÃO

**Deteção e seguimento de objetos
através de câmaras de profundidade**



Autores: Luís Rei 78486
Gonçalo Duarte 78675
João Girão 78761

23 de Dezembro de 2017

Conteúdo

1	Introdução	1
1.1	Câmera Kinect	1
1.2	Modelo da câmera	2
1.3	Determinação de pontos-chave	4
1.4	Correspondência de objetos	4
1.5	Random Sample Consensus	5
1.6	Procrustes	6
2	Implementação	6
2.1	Obtenção da imagem de fundo	6
2.2	Distinção de objetos	6
2.3	Identificação de um objeto singular	7
2.4	Seguimento de objetos entre frames	8
2.4.1	Análise de correspondência	8
2.4.2	Histograma da cor	9
2.4.3	Excentricidade	9
3	Experimentação	9
3.1	Seguimento de objetos	10
3.1.1	Dataset <i>duascamaras</i>	10
3.1.2	Dataset <i>maizena chocapic</i>	10
3.2	Aproximações usando o RANSAC	11
3.2.1	Dataset <i>lab1</i>	11
3.2.2	Dataset <i>maizena chocapic</i>	11
3.2.3	Dataset <i>room1</i>	12
3.2.4	Dataset <i>duascamaras</i>	12
4	Conclusões	12

1 Introdução

O principal objetivo deste projeto é a deteção e seguimento de objetos a partir de imagens RGB e medições de profundidade extraídas de câmaras 3D que se encontram em pontos distintos de um mesmo cenário. Para alcançar este feito, as informações provenientes das câmaras estáticas são fundidas por forma a ser construída uma visão mais abrangente de todo o ambiente (figura 1).

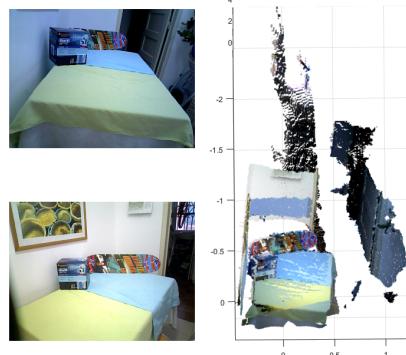


Figura 1: Fusão das imagens de RGB+profundidade para construção de cenário mais completa.

Alinhando as representações locais oriundas das nuvens de pontos 3D, o modelo do cenário poderá ser construído a partir dos pontos emparelhados. A posição e orientação das câmaras Kinect, em relação a um único referencial, é calculada através de uma análise das sequências de imagens disponibilizadas. A distinção entre objetos comuns às duas câmaras e exclusivos a uma só é feita através da comparação das projeções 3D dos objetos num referencial comum, produto da fusão das imagens de profundidade de ambas as câmaras. O seguimento de um objeto é feito com base na informação extraída na *frame* anterior.

Para conseguir detetar movimento, e dada uma sequência de imagens de profundidade, as tarefas a cumprir são as seguintes:

1. Calcular o plano de fundo da sequência de imagens de profundidade;
2. Extrair para cada fotograma o primeiro plano com base no plano de fundo obtido anteriormente (diferença entre profundidade de pontos maior que um certo valor);
3. Filtrar a imagem com base no gradiente de profundidade e filtrar a legendagem do primeiro plano com base no tamanho dos objetos.

Para acompanhar o movimento dos objetos a linha de ação a tomar é a seguinte:

1. Identificação de pontos-chave nas duas *frames*;
2. Análise probabilística da correspondência entre as *features* extrapoladas;
3. Adequação do *matching* com base em informação recolhida desde o início da análise sequencial.

1.1 Câmara Kinect

As câmaras Kinect foram desenvolvidas para permitir que o utilizador consiga controlar e interagir, através de gestos e movimentos, com um computador ou consola. O Kinect é composto por dois tipos de câmaras distintos: uma câmara de cor e uma câmara de profundidade. O seu modelo geométrico encontra-se na figura 2.

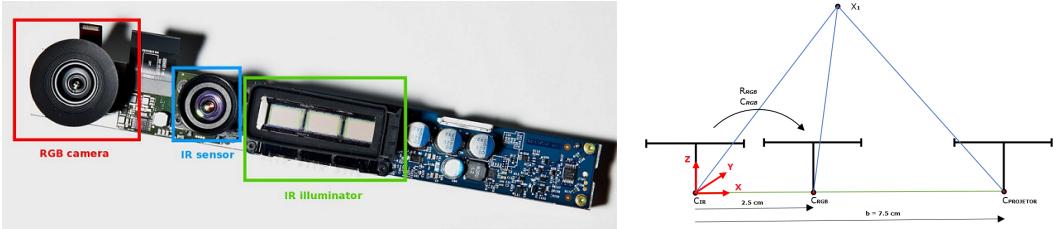


Figura 2: Câmara Kinect (esquerda) e correspondente modelo geométrico (direita).

Estas duas câmeras possuem a mesma resolução e permitem a obtenção de dois tipos de imagens - uma imagem de cor RGB e outra de profundidade (figura ??). Combinadas, permitem a reconstrução de um cenário 3D a partir da associação dos valores RGB de um pixel, para o pixel correspondente na imagem de profundidade.



Figura 3: Imagem RGB (esquerda) e correspondente imagem de profundidade (direita).

1.2 Modelo da câmara

Cada uma das câmeras referidas projeta uma conjunto de pontos do mundo real em 3D (representado por $\mathbf{X} = [X \ Y \ Z]^T$) para um plano de imagem com uma dimensão inferior onde os pontos são dados por $\mathbf{x} = [x \ y]^T$ (figura 4).

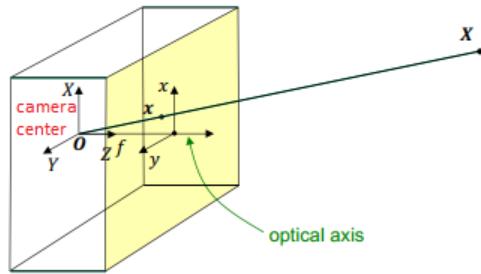


Figura 4: Modelo da câmara

Adotando a notação acima, e considerando uma câmera com uma distância entre o centro de projeção e o ponto de projeção unitária (distância focal unitária), a projeção da perspetiva é dada por $x = \frac{X}{Z}$, $y = \frac{Y}{Z}$.

As câmeras modernas compensam a inversão da imagem produzida por uma câmera que segue o modelo *pin-hole* fazendo uma projeção num plano frontal.

Em coordenadas homogéneas a projeção é dada por

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad \lambda \tilde{x} = [I \ 0], \tilde{X} = X. \quad (1)$$

Generalizando a especificação feita anteriormente, temos modelos mais complexos caracterizados por parâmetros intrínsecos (ponto principal, fator de escala, foco) e extrínsecos (localização arbitrária de um referencial mundo).

O modelo interno define um novo sistema de coordenadas definido por uma transformação 2D (equações 2 e 3), onde é feita uma conversão de unidades métricas para pixels.

$$x' = fs_x x + c_x, \quad (2)$$

$$y' = fs_y y + c_y, \quad (3)$$

onde temos que x e y estão em coordenadas métricas, x' e y' estão em pixels, f é a distância focal, c_x e c_y são as coordenadas dos pontos principais [pixels], e s_x e s_y são os factores de escala nas direções de x e de y respetivamente [pixel/m].

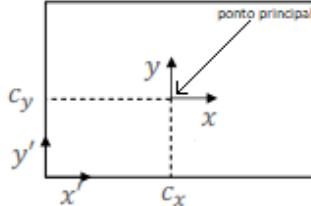


Figura 5: Modelo interno.

As coordenadas homogéneas vistas em 1 são então descritas por

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} fs_x & 0 & c_x \\ 0 & fs_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad \tilde{x}' = K\tilde{X}, \quad (4)$$

onde K é a matriz triangular superior que alberga os parâmetros intrínsecos.

Em relação aos parâmetros extrínsecos temos o referencial do mundo que é definido por uma transformação 3D

$$X = RX' + T, \quad (5)$$

onde temos que X é o referencial da câmara, X' o referencial do mundo, $R \in \mathbb{R}^{3 \times 3}$ uma matriz que dita a rotação entre a câmara e o referencial do mundo, $T \in \mathbb{R}^3$ um vector que representa a origem do referencial do mundo nas coordenadas da câmara.

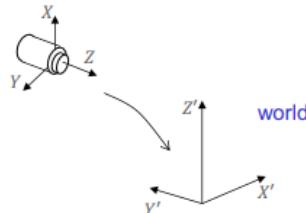


Figura 6: Modelo externo.

Temos que em coordenadas homogéneas:

$$\tilde{X} = \begin{pmatrix} R & T \\ 0^T & 1 \end{pmatrix} \tilde{X}' \quad (6)$$

Combinando a equação 6 com a 1 temos:

$$\lambda \tilde{x} = (R \ T) \tilde{X}' \quad (7)$$

Adicionalmente, e considerando 4, o modelo final torna-se

$$\lambda \tilde{x}' = K(R \ T) \tilde{X}' = P \tilde{X}', \quad (8)$$

onde P é a matriz 3×4 característica da câmara.

Com esta informação é possível combinar os dados de cor e profundidade numa só imagem que acumula toda a informação (figura 7).



Figura 7: Junção das informações das duas câmaras - profundidade e RGB - para imagens de um mesmo *dataset*.

1.3 Determinação de pontos-chave

A deteção e descrição de *features* locais de imagens pode auxiliar no reconhecimento de objetos. Estes pontos definem o que é interessante a imagem tem e são caracterizados por terem pontos vizinhos com variações de intensidade em mais do que uma direção (figura 8).

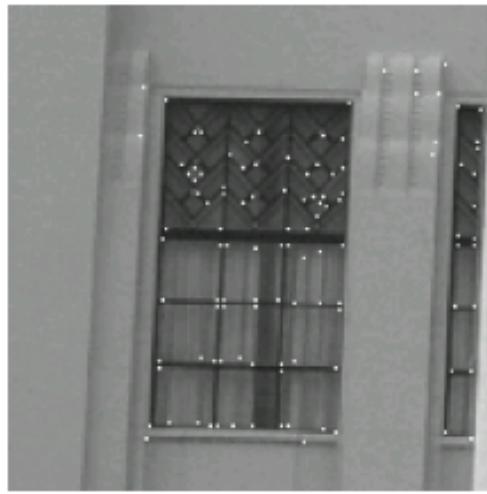


Figura 8: Pontos-chave de uma imagem.

Pontos invariantes a escalamento de uma imagem ou a rotações, robustos a mudanças de iluminação, ruído e ligeiras mudanças de perspetiva, permitem a identificação de um objeto com uma probabilidade de erro de *matching* mais pequena devido a serem descritos sempre da mesma forma. O método Scale-Invariant Feature Transform (SIFT) é um algoritmo que deteta e extrai os denominados pontos-chave de uma imagem a partir da diferença de Gaussianas (9). Tem também várias medidas para evitar pontos que não sejam ideais para detetar *features*, sendo que elimina pontos com pouco contraste e pontos que estão na borda da imagem.

$$D(x, y, \sigma) = G(x, y, \sigma) * |(x, y) - G(x, y, k\sigma) * |(x, y)| \quad (9)$$

Bastam 3 *features* de um objeto para que o SIFT consiga calcular a sua localização, posição e orientação. Um detetor de pontos-chave extraí de uma *frame* um conjunto de regiões consistentes com algumas variações de luminosidade, perspetiva e outras condições de visualização. O descriptor associa a uma dada região uma assinatura que identifica a aparência de forma compacta e robusta.

1.4 Correspondência de objetos

O seguimento de objetos numa sequência de imagens é feito através de associações entre objetos numa imagem e na imagem anterior. Um objeto é reconhecido na nova imagem comparando individualmente cada *feature* da nova imagem com as anteriores, encontrando um conjunto de pontos-chave candidatos baseado na norma euclidiana dos seus vetores de *features*. Estas associações têm 3 possíveis resultados: $x_{i-1} \leftrightarrow x_i(a), \emptyset \rightarrow x_i(b)$ ou $x_{i-1} \rightarrow \emptyset(c)$. No caso (a) temos que o objeto x na imagem $i-1$ corresponde ao

objeto x na imagem i , enquanto que nos casos (b) e (c) o objeto x nasce ou morre, respectivamente. Os casos (b) e (c) são triviais quando uma das imagens a analisar não possui nenhum objeto, mas torna-se menos fácil distinguir entre os três casos quando ambas as imagens têm objetos.

De modo a ultrapassar este problema construiu-se uma matriz A que relaciona os objetos em ambas as imagens, resultante da união de três outras matrizes M, C e E . As quatro matrizes têm dimensões $(n + 1) \times (m + 1)$, onde n e m são o número de objetos na imagem anterior e atual, respectivamente. A entrada a_{ij} nesta matriz A deve ser lida como a probabilidade do objeto i na imagem anterior ser o objeto j da imagem atual, ou de nascer se $i = n + 1$ e de morrer se $j = m + 1$ (a entrada $a_{(n+1)(m+1)}$ não deve ser considerada visto que um objeto que nasce e morre entre imagens consecutivas não é detetado). A matriz M é chamada a matriz de correspondência e é feita com as features retiradas do SIFT, onde cada entrada é dada por $\frac{\#matches}{\#features}$. A matriz C é criada com a exponencial negativa das diferença de histogramas de cor e, semelhantemente, a matriz E com a diferença de excentricidade (em 2D) entre objetos de duas imagens.

A matriz A vem então como:

$$A = \alpha M + \beta C + \gamma E \quad , \alpha = 0.6, \beta = 0.2, \gamma = 0.2 \quad (10)$$

Note-se que como se trata de uma matriz de probabilidades onde cada linha deve somar 1, a entrada da última coluna da linha j terá um valor correspondente a $1 - \sum_{i=1}^m a_{ij}$.

Procede-se então a uma procura *greedy* na matriz, começando na primeira linha, escolhendo a coluna que maior probabilidade tem e, se esta não for a última, bloqueando-a para que não seja novamente escolhida. Caso dois objetos sejam relacionados através desta procura, a informação do objeto da imagem atual será adicionada à restante informação do objeto correspondente. No fim da procura, todos os objetos que não foram relacionados ou morreram serão considerados objetos novos.

1.5 Random Sample Consensus

Após os pontos chave serem escolhidos e emparelhados é possível estimar a relação entre os conjuntos de pontos e calcular a matriz de rotação R e o vetor de translação T . O método RANSAC é baseado na geração de hipóteses e classificação de pontos como *inliers* e *outliers*, e consiste num processo iterativo de estimativa de parâmetros que melhor se adequem a um modelo a partir de um conjunto de observações que contém *outliers* (observações que deveríamos rejeitar e não contabilizadas). O algoritmo é composto pelos seguintes passos:

1. Selecionar um modelo que se adeque aos dados.
Modelo dado por $p' = Rp + T$.
2. Selecionar aleatoriamente uma amostra com o número mínimo de pontos necessários para determinar o modelo.
Uma transformação projetiva necessita de 8 pontos para determinar os parâmetros do modelo (4 pares de pontos-chave).
3. Usar estes pontos no método para encontrar o modelo. *Determinamos R_i e T_i resolvendo o problema procrustes para cada iteração.*
4. Verificar o número de pontos do dataset que são consistentes com este modelo. Contar o número de inliers e outliers.
Calcular o erro $e = ||p'_k - R_i P_k - T_i||$, onde k são o par de pontos considerados. Se $e < threshold$ os pares de pontos são considerados inliers.
5. Repetir os passos 2-4 para um certo número de iterações.
6. Considerar o modelo para o qual foi detetado um maior número de *inliers*.

1.6 Procrustes

O problema de Procrustes é usado para determinar as matrizes de rotação e translação óptimas de um objeto em relação a outro. Este algoritmo recebe um conjunto de pontos p_n e p_n' com a mesma dimensão, assumindo que existe uma relação inerente entre ambos. Estes pontos estão relacionados por

$$p'_n = Rp_n + T + V_n, \quad (11)$$

onde $R \in \mathbb{R}^{3x3}$ é uma matriz de rotação, $T \in \mathbb{R}$ é um vector de translação e $V_n \in \mathbb{R}$ é um vetor de ruído.

Para encontrar R , T óptimos por forma a mapear os pontos p_n em p_n' é necessário resolver um problema de minimização dado pelo critério do erro quadrado que se apresenta como

$$\sum_{i=0}^N ||p'_n - Rp_n - T||^2. \quad (12)$$

Sabe-se que os centróides são o ponto onde a distância a todos os outros pontos é menor. Como tal tem-se

$$\bar{p}' = \frac{1}{N} \sum_{i=0}^N p'_n, \quad p'_c = p'_n - \bar{p}', \quad (13)$$

$$\bar{p} = \frac{1}{N} \sum_{i=0}^N p_n, \quad p_c = p_n - \bar{p}. \quad (14)$$

Os pontos p_n e p_n' têm o mesmo centróide e como tal pode simplificar-se o problema anterior para

$$\sum_{i=0}^N ||p'_c - Rp_c||^2. \quad (15)$$

Para se encontrar o T , basta manipular as equações anteriores e substituir em

$$T = \bar{p}' - R\bar{p}. \quad (16)$$

2 Implementação

2.1 Obtenção da imagem de fundo

Para identificar os vários objetos é necessário separar o plano de fundo do primeiro plano em cada imagem. Para obter o background utilizou-se um filtro com base na mediana nas primeiras 25 imagens de profundidade, de modo a poupar memória. Optou-se por este filtro em vez da média visto que o primeiro seleciona, para cada ponto, um valor de profundidade existente e é mais resistente ao ruído, pois elimina ou suaviza picos de valores. Usaram-se as imagens em profundidade em vez de imagens RGB, porque devido à mudança de iluminação, mesmo que pequena, as imagens de cor tendem a captar valores mais diferentes de *frame* para *frame* que as de profundidade. Apesar de tudo, para espaços com considerável profundidade (6 metros) os erros obtidos crescem bastante de dimensão.

2.2 Distinção de objetos

Como a distinção de objetos é efetuada em 2D temos que nos precaver perante a possibilidade de dois objetos (elementos distantes o suficiente para serem considerados diferentes) se encontrarem de alguma forma sobrepostos na imagem (figura 9).

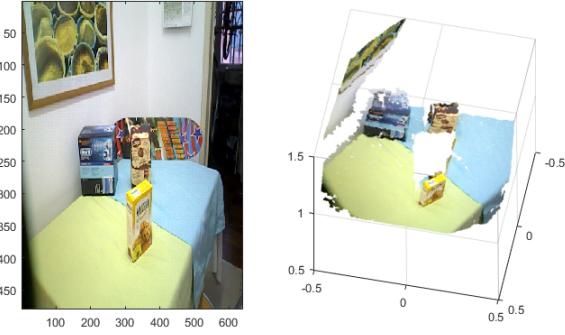


Figura 9: Projeção da imagem RGB (esquerda) para o plano 3D (direita).

Para tal, aplicamos uma máscara lógica de gradiente efetuada a partir da imagem original, permitindo a deteção de discrepâncias de profundidades superiores a 20 cm (figura 10).

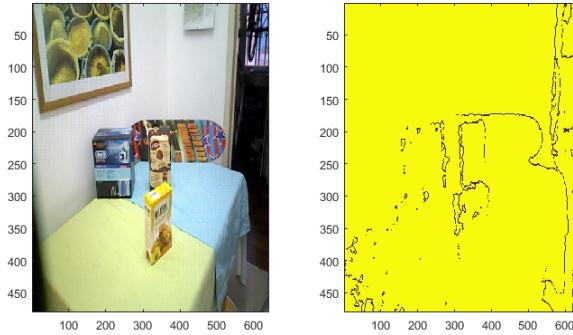


Figura 10: Imagem RGB original (esquerda) e correspondente máscara de gradiente lógica para diferenças de profundidade acima dos 15 cm (direita).

Na figura 11 vê-se o efeito que o método do gradiente tem na imagem, sendo que estão destacados todos os pontos considerados importantes pelo método, pontos que podem ser considerados relevantes para encontrar objectos. No segundo conjunto de imagens é possível ver os contornos dos objectos identificados. Apenas a segunda imagem usa o método do gradiente e como tal consegue distinguir três objectos em vez de apenas dois (a imagem anterior junta dois objectos num só).

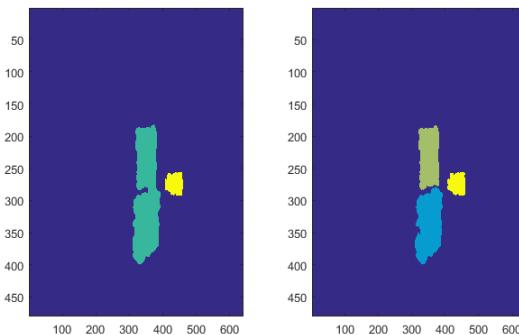


Figura 11: Legendagem da *frame* antes e depois de aplicar a filtragem correspondente à máscara de gradiente.

2.3 Identificação de um objeto singular

Antes de analisar o objeto num espaço de dimensão 3 recorremos ao *downsample* para diminuir o esforço computacional associado ao cálculo do centróide correspondente a uma dada nuvem de pontos. O resultado do *downsample* é visível na figura 12.

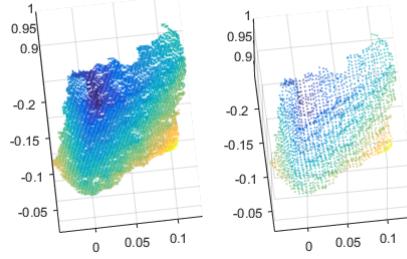


Figura 12: Exemplo do efeito de *downsampling*. As características físicas do objeto projetado em 3D são mantidas apesar da redução do número de pontos de representação.

Após a redução de pontos necessários para representar os elementos em análise determinamos o centróide como a média da nuvem de pontos, e comparamos a distância euclidiana entre as *pointclouds* unidas e predispostas sobre um mesmo referencial. Objetos distantes a mais de 20 cm foram considerados instâncias de objetos diferentes.

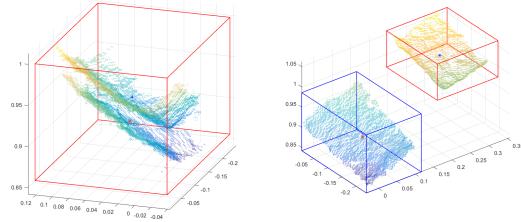


Figura 13: Distinção entre a representação de um único (esquerda) e dois disjuntos objetos (direita) com base na norma euclidiana dos centróides.

2.4 Seguimento de objetos entre frames

O seguimento de objetos entre frames é feito com base em três métodos distintos, cada um com um peso associado. O peso associado à análise de correspondência é maior que o dos restantes métodos devido ao facto deste ser mais robusto no que toca ao ruído.

2.4.1 Análise de correspondência

Para encontrar correspondências usa-se o método SIFT, referido anteriormente, no *foreground* das imagens.

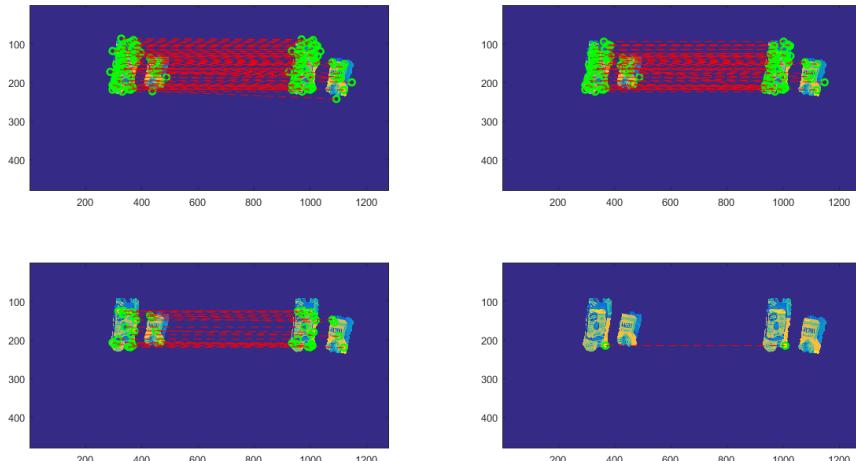


Figura 14: Variação do número de features extraídas com a variação do parâmetro *peakthresh*. O parâmetro assumiu o valor de 0 (superior esquerdo), 0.03 (superior direito), 0.06 (inferior esquerdo) e 0.7 (inferior direito).

2.4.2 Histograma da cor

A cor é analisada com base numa escala HSV (hue, saturation, value). Isto para evitar que a intensidade/saturação de um determinado pixel seja mal analisado devido a mudanças de luminosidade.

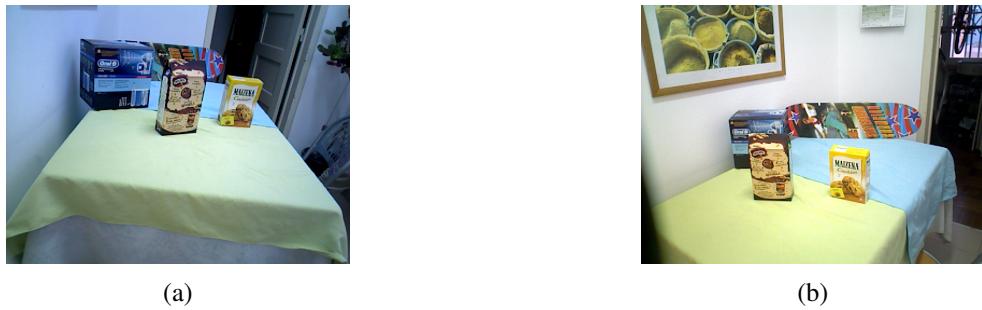


Figura 15: Imagens RGB correspondentes aos histogramas

A diferença dos histogramas tem como base a distância euclidiana $\sum_k [P_1(k) - P_2(k)]^2$. Por forma a que o histograma de um pixel caracterizado pela cor preto se transponha para uma contagem de branco, ao último intervalo da escala RGB é subtraído 1 (histogramas têm intervalos de RGB de dois valores).

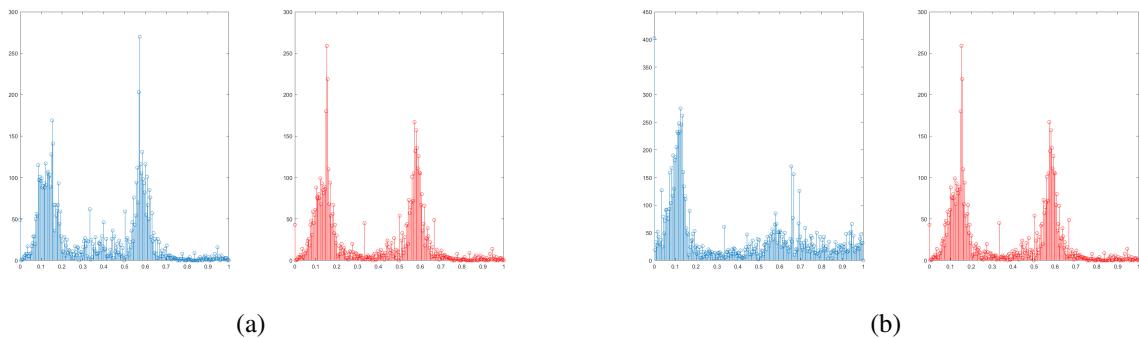


Figura 16: Análise do histograma de cor de dois objetos iguais (esquerda) e dois objetos distintos (direita).

2.4.3 Excentricidade

É feita ainda uma análise sobre a forma dos objetos a partir do estudo da matriz de covariância que caracteriza o objeto. Depois de recentrado o referencial no centro dos pontos, e normalizada a matriz, são calculados os valores próprios e respetivos vetores próprios. A um maior valor próprio está associada uma maior variância dos dados.

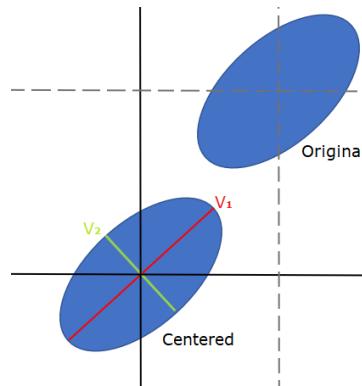


Figura 17: Excentricidade de um objeto explicada por vetores próprios que traduzem variância dos dados.

3 Experimentação

Secção onde são apresentados diversos resultados experimentais.

3.1 Seguimento de objetos

3.1.1 Dataset *duascamaras*

O plano de fundo do dataset para a câmara 2 tem um vestígio de uma garrafa (um dos objetos a seguir), aspecto que influencia o número de objetos a apresentar no final do algoritmo.



Figura 18: Background do dataset duascamaras



Figura 19: Imagens do Dataset duas câmaras (1).



Figura 20: Imagens do Dataset duas câmaras (2).

Nas imagens anteriores é possível ver as labels dos objectos de frame para frame, sendo de notar que apenas é marcado as zonas onde ocorre movimento, ou seja, objectos que foram deslocados de um frame para o seguinte. Na câmara dois é de notar que aparece um objecto que não se encontra na câmera 1, uma perna de alguém que estava na sala no momento de obtenção das imagens.

3.1.2 Dataset *maizena chocapic*

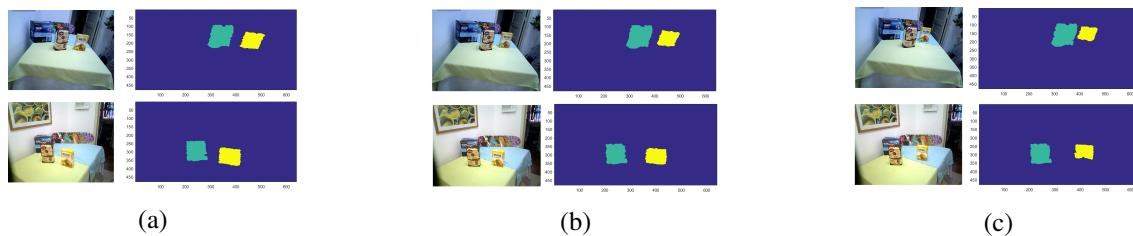


Figura 21: Seguimento de objetos no dataset especificado.

Neste dataset em particular é possível ver que há dois objectos em movimento de frame para frame, como mostram os labels. É de notar que o algoritmo não confunde ou perde objectos de um frame para o seguinte.

3.2 Aproximações usando o RANSAC

3.2.1 Dataset *lab1*

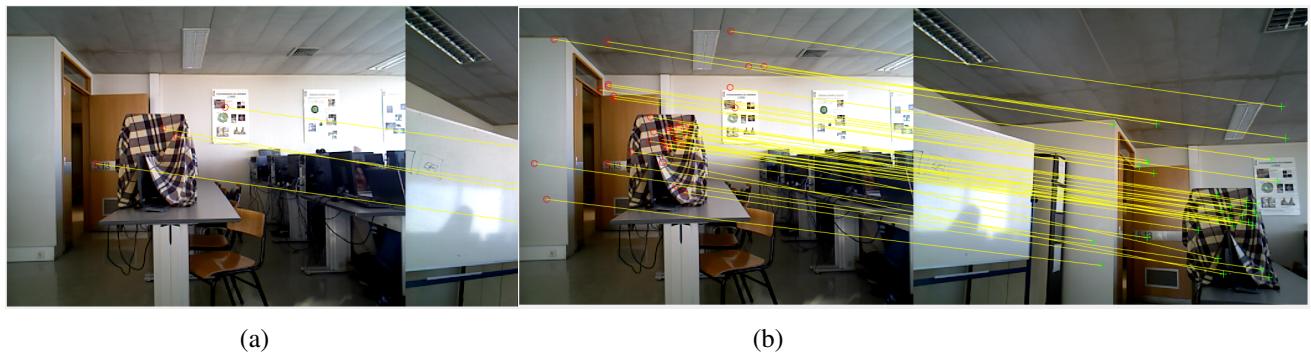


Figura 22: *Keypoints* que culminaram no maior número de inliers (esquerda) e todos os inliers detetados com base no valor de threshold de 0.3 definido (direita).



Figura 23: Point cloud do dataset lab1

3.2.2 Dataset *maizena chocapic*

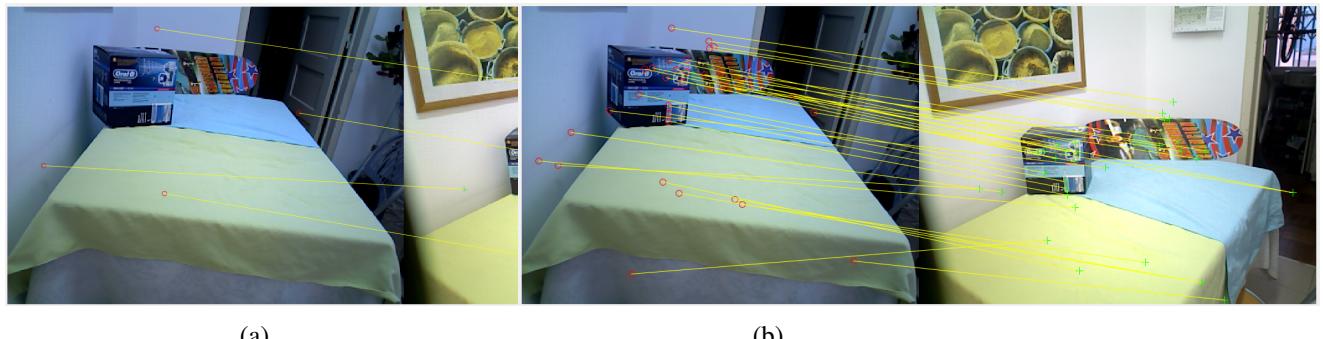


Figura 24: *Keypoints* que culminaram no maior número de inliers (esquerda) e todos os inliers detetados com base no valor de threshold de 0.3 definido (direita).

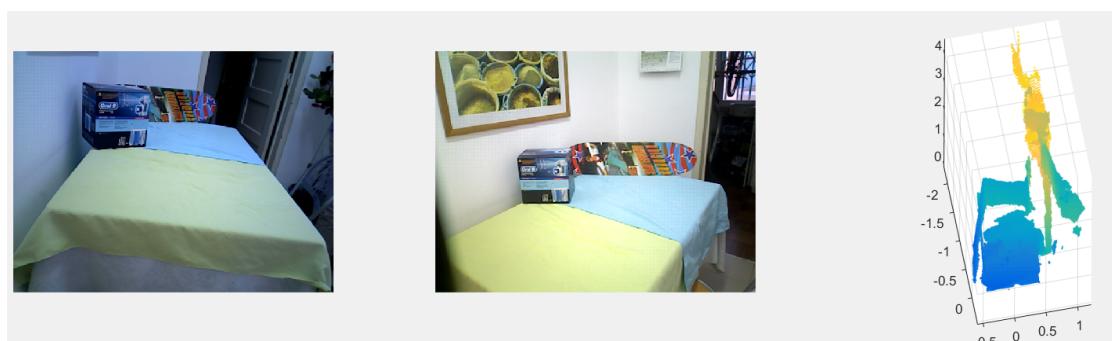


Figura 25: Point cloud do dataset maizena chocapic

3.2.3 Dataset room1

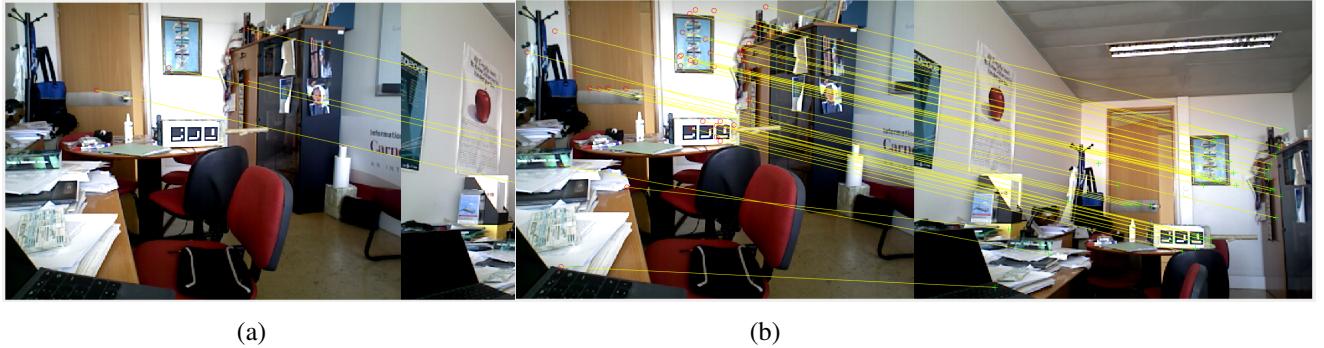


Figura 26: *Keypoints* que culminaram no maior número de inliers (esquerda) e todos os inliers detetados com base no valor de threshold de 0.3 definido (direita).

3.2.4 Dataset duascamaras



Figura 27: *Keypoints* que culminaram no maior número de inliers (esquerda) e todos os inliers detetados com base no valor de threshold de 0.3 definido (direita).

4 Conclusões

O projeto permitiu explorar um conjunto de técnicas de processamento de imagem e visão, assim como ajudar a compreender as dificuldades associadas a problemas deste tipo e a desenvolver o pensamento crítico necessário para a resolução destes.

O RANSAC provou-se eficaz em aproximar conjuntos de nuvens de pontos a partir do maior número de inliers possível. Houve no entanto algumas dificuldades em datasets com padrões repetidos.

Em relação aos resultados obtidos a maioria destes foram positivos, principalmente na primeira fase do projecto onde não se usa o RANSAC, visto que o algoritmo consegue detetar e localizar objectos de forma consistente para todos os datasets usados.