Congreso de Métodos Numéricos en Ingeniería 2005
Granada, 4 a 7 de Julio, 2005

OBTENÇÃO DA FORMA 3D DE OBJECTOS USANDO UMA METODOLOGIA DE RECONSTRUÇÃO DE ESTRUTURAS A PARTIR DO MOVIMENTO

Teresa C. S. Azevedo [1], João Manuel R. S. Tavares [1,2*] e Mário A. P. Vaz [1,2]

1: Laboratório de Óptica e Mecânica Experimental,
Instituto de Engenharia Mecânica e Gestão Industrial
Rua Dr. Roberto Frias s/n, 4200-465 Porto, Portugal
e-mail: {teresa.azevedo, tavares, gmavaz}@fe.up.pt, web: http://www.inegi.up.pt

2: Departamento de Engenharia Mecânica e Gestão Industrial,
Faculdade de Engenharia da Universidade do Porto
Rua Dr. Roberto Frias s/n, 4200-465 Porto, Portugal
e-mail: {tavares, gmavaz}@fe.up.pt, web: http://www.fe.up.pt

Palavras-chave: Visão computacional, Visão activa, Reconstrução 3D, Estrutura a partir do movimento

Resumo. *Em Visão Computacional, várias são as técnicas sem contacto e sem projecção de energia que se podem utilizar para recuperar a estrutura 3D de uma cena ou de um objecto. O objectivo principal do projecto a reportar neste artigo, é a obtenção de modelos 3D definidores da geometria de objectos reais, utilizando metodologias de Reconstrução de Estruturas a partir do Movimento da(s) câmara(s) ou dos próprios objectos. Este procedimento é habitualmente designado por "obtenção da estrutura a partir do movimento", e no caso presente, não se pretende impor qualquer tipo de restrição ao movimento. Assim, partindo de uma sequência de imagens não calibradas, pretende-se extrair o movimento em causa, a calibração da(s) câmara(s) considerada(s), e obter a geometria 3D do objecto em causa. Resumidamente, neste artigo são apresentadas as metodologias mais usuais neste domínio, descrito o trabalho por nós já desenvolvido, apresentados alguns resultados experimentais obtidos e respectivas análises e, por último, serão enumeradas algumas conclusões e perspectivas de trabalho futuro.*

1. INTRODUÇÃO

O presente artigo, tem como principais objectivos analisar e descrever resumidamente o estudo realizado sobre cinco *softwares* e uma biblioteca computacional, todos de domínio público, para a Reconstrução Tridimensional de Objectos usando técnicas de Visão Activa. Tais entidades serão neste artigo referidos genericamente como Programas 1 a 6 :

- Programa 1 – *Peter's Matlab Functions for Computer Vision and Image Analysis* [1]: pacote de funções em *Matlab* de Visão 3D e processamento e análise de imagens;
- Programa 2 – *Torr's Matlab Toolkit* [2]: com uma interface gráfica em *Matlab*, usa o procedimento de "estrutura a partir do movimento" em duas imagens;
- Programa 3 – *OpenCV* [3]: biblioteca de funções em C^{++} que implementam alguns dos algoritmos mais usuais no domínio da Visão Computacional;
- Programa 4 – *KLT* [4]: conjunto de funções em *C*, que implementam o algoritmo de *Lucas-Kanade-Tomasi* [5, 6];
- Programa 5 – *Projective Rectification without Epipolar Geometry* [7]: programa em *C*, que realiza a rectificação de duas imagens estéreo sem a pré-determinação da geometria epipolar;
- Programa 6 – *Depth Discontinuities by Pixel-to-Pixel Stereo* [8]: programa em *C*, que retorna os mapas de disparidade e descontinuidades entre duas imagens rectificadas.

Ao longo deste trabalho, desenvolveu-se uma aplicação em C^{++}, com interface gráfica adequada para o utilizador, onde se integraram algumas das funções disponibilizadas nos programas referidos, de forma a analisar comparativamente o desempenho de cada. Para os testes experimentais, utilizaram-se imagens de dimensão 540×612. As funções analisadas abrangem várias técnicas de Visão 3D: desde a extracção de pontos fortes, ou seja pontos característicos, emparelhamento desses pontos entre imagens, cálculo da geometria epipolar, rectificação e emparelhamento denso (Figura 1).
Após a descrição das técnicas de Visão 3D analisadas, é apresentada uma breve descrição da sua implementação, nos vários programas considerados, são apresentados alguns resultados obtidos pelos mesmos, as conclusões finais e indicadas as perspectivas de trabalho futuro.

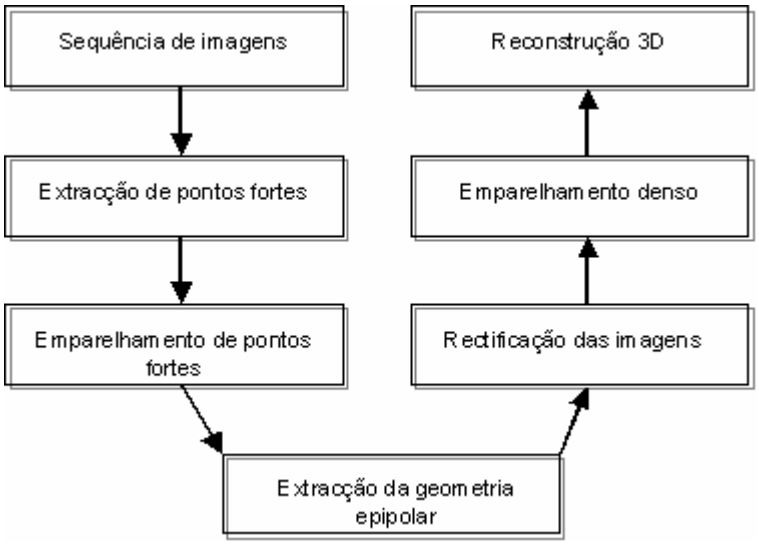

Figura 1. Sequência das operações analisadas para a reconstrução 3D de objectos suando Visão Activa.

2. EXTRACÇÃO DE PONTOS FORTES

O primeiro passo da maioria das técnicas usuais de Visão 3D, é a detecção de pontos fortes numa imagem. Esses pontos são aqueles que têm uma componente 2D característica sobre o objecto (usualmente, são os vértices dos objectos). A sua extracção, permite posteriormente correlacioná-los sequencialmente noutras imagens, e este emparelhamento torna possível a posterior reconstrução 3D.

2.1. Detector de vértices de *Harris*

O método de *Harris* foi proposto em 1988, por *Harris* e *Stephens* [9], e considera o mínimo e o máximo valor próprio, α e β respectivamente, da matriz de covariância do gradiente de uma imagem binária:

$$M = \begin{bmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) \\ \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) & \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix}, \tag{1}$$

onde $(\partial I/\partial x)$ e $(\partial I/\partial y)$ são os gradientes da imagem I nas direcções x e y, respectivamente. Quando os dois valores próprios são elevados e similares em magnitude, significa que o gradiente tem uma estrutura bidimensional característica, ou seja, foi detectado um vértice. Por forma a evitar uma decomposição explícita dos valores próprios de M, *Harris* definiu a seguinte função de medida:

$$R = \det(M) - k(trace(M))^2, \tag{2}$$

onde $\det(M) = \alpha\beta$ e $trace(M) = \alpha + \beta$. *Harris* sugere $k = 0.04$, mas como usualmente este parâmetro necessita de ser estimado, *Noble* [10] sugeriu um re-arranjo na função de medida:

$$R = \det(M)/(trace(M) + \varepsilon). \tag{3}$$

Se os valores próprios forem similares e elevados, $\det(M)$ é muito maior que $trace(M)$, logo o resultado R será também elevado. Já ε, é uma constante ($2\text{x}10^{-52}$) utilizada por forma a evitar um denominador nulo no caso de M ter traço nulo.

Por forma a minimizar os resultados obtidos, pode-se impor que R seja superior a um dado valor de *threshold*. Como este método é muito sensível ao ruído, é aplicado aos termos de M um filtro de suavização Gaussiano com um desvio padrão σ. No entanto, isto introduz uma redução na precisão da localização obtida para os pontos fortes. Para compensar este facto, aplica-se o operador morfológico de dilatação, com uma janela de tamanho r. Adicionalmente, pode-se rejeitar vértices muito próximos: dentro de uma certa área de raio d, escolhe-se o vértice mais forte.

2.2. Métodos de extracção de pontos fortes utilizados pelos programas

Os programas 1 e 2, utilizam o método de *Harris* para a detecção de vértices. Ambos calculam a matriz de covariância M, utilizando o operador de gradiente *Prewitt*. O programa 1, utiliza (3) para definir a função de medida R e determina como pontos fortes (x, y) aqueles que satisfaçam a condição de *threshold* $R(x, y) \geq t$ e a condição $R(x, y) = D(x, y)$, sendo D a imagem dilatada de R. Já o programa 2, usa (2), com $k = 0.04$, elimina todos os pixéis que não forem máximos locais, numa janela 3×3 e selecciona os n pontos mais fortes. Os programas 3 e 4, determinam os pontos fortes directamente a partir da matriz de covariância M: calculam dos mínimos valores próprios de M e eliminam os pontos (x, y) que não satisfaçam a condição de *threshold* $M(x, y) \geq t$. O programa 3, selecciona o vértice mais forte, dentro de uma área de raio d, que é aquele que tiver o maior valor de M, e que satisfaça também a condição $M(x, y) = D(x, y)$. O programa 4 selecciona o vértice mais forte, dentro de uma janela $d \times d$ e no final selecciona os n pontos mais fortes.

3. EMPARELHAMENTO DE PONTOS ENTRE IMAGENS (*MATCHING*)

Na reconstrução 3D, é necessário identificar os pontos nas várias imagens que resultem da projecção do mesmo ponto da cena (*matching*). Um pequeno número de pontos de correspondência, é suficiente para se poder determinar uma relação geométrica entre duas imagens.

3.1. Correlação simples

Este método é conhecido como *SSD* (*sum-of-square-differences*). Considere-se o ponto $m = [m_x \quad m_y]$ da imagem I. O objectivo é encontrar o ponto $m' = m + d = [m_x + d_x \quad m_y + d_y]$ na imagem J tal que $I(m)$ e $J(m')$ sejam pontos similares. Tal obtém-se minimizando a função de dissimilaridade, definida como:

$$D(d) = \iint_w (I(x, y) - J(x + d_x, y + d_y))^2 . \tag{4}$$

Note-se que esta medida é efectuada numa região dentro da imagem de tamanho w, denominada por janela de integração.

3.2. Correlação normalizada de média nula

A medida de dissimilaridade apresenta algumas restrições, principalmente no facto de não ser invariante a mudanças de intensidade global ou de contraste nas regiões de interesse.
Estas dificuldades, são superadas se se normalizarem os vectores referentes às imagens, resultando no seguinte coeficiente de correlação:

$$S(d) = \frac{\iint_w (I(x,y) - \bar{I}).(J(x+d_x, y+d_y) - \bar{J}))}{\sqrt{\iint_w (I(x,y) - \bar{I})^2} . \sqrt{\iint_w (J(x+d_x, y+d_y) - \bar{J}))^2}}, \quad (5)$$

onde $\bar{I}$ e $\bar{J}$ são os valores médios nas regiões de interesse definidas por w. Este método é denominado por *ZNCC* (*zero-mean normalized cross-correlation*).

3.3. Algoritmo de *Lucas-Kanade*

Este algoritmo assume que, se o vector de deslocamento $d = [d_x \quad d_y]$ for reduzido, prova-se que a minimização de (4) se resume a resolver o sistema de duas equações a duas incógnitas:

$$d = e.G^{-1}, \quad (6)$$

onde G é a matriz de covariância do gradiente da primeira imagem e e é a diferença entre as duas imagens ao longo de G.
Este algoritmo, é utilizado de uma forma iterativa: começa-se por obter uma estimativa de d a partir de imagens suavizadas de baixa resolução (imagens piramidais), refinando os pontos de emparelhamento com imagens de resolução cada vez mais elevada, até que na última iteração se utilizam as imagens originais [5, 11].

3.4. Métodos de emparelhamento de pontos fortes utilizados pelos programas

O programa 1, utiliza a medida de similaridade do método *ZNCC*, dada por (5), e o programa 2, a medida de dissimilaridade do método *SSD*, dada por (4). Utilizando os pares de pontos fortes (m, m') obtidos nas duas imagens estéreo, aplicam a função de medida para cada par que satisfaça a condição $\|m - m'\| \leq d^2$, e seleccionam aqueles que, em ambas as direcções, $\overrightarrow{mm'}$ e $\overrightarrow{m'm}$, maximizam (5) ou minimizam (4), consoante o caso.
Por outro lado, os programas 3 e 4 utilizam o método de *Lucas-Kanade*, que necessita somente do conjunto de pontos fortes m encontrados na primeira imagem, do número de imagens piramidais n e do número de iterações para tentar encontrar d em cada ponto m.
No programa 4, a minimização de (6) é realizada pelo método iterativo de *Newton-Raphson* [5], que é interrompido se ocorrer um destes casos:

I. o ponto calculado sofre um desvio superior a max_d;
II. o determinante da matriz de gradiente é inferior a min_det;
III. o número de iterações excede n;
IV. o ponto encontra-se fora da imagem;
V. o resíduo e é maior que max_e.

No programa 3, o processo iterativo é interrompido nos casos II, III e IV [11].

4. GEOMETRIA EPIPOLAR

O cálculo da geometria epipolar significa não só obter alguma informação da pose relativa entre duas vistas sobre a mesma cena, como também eliminar erros no emparelhamento efectuado previamente, e obter novos pontos de correlação [12]. A geometria epipolar corresponde à estrutura geométrica entre duas vistas, e expressa-se matematicamente pela matriz fundamental F, tal que:

$$m'^T F m = 0\,, \tag{7}$$

onde m e m' são os pontos de correspondência entre duas imagens. O ponto m, passa pela linha epipolar l, definida por:

$$l = Fm'\,. \tag{8}.$$

4.1. Algoritmo de 8 pontos

Re-arranjando (7) da seguinte forma:

$$[xx' \quad yx' \quad x' \quad xy' \quad yy' \quad y' \quad x \quad y \quad 1]F = 0\,, \tag{9}$$

com $m = [x \quad y \quad 1]^T$ e $m' = [x' \quad y' \quad 1]^T$, então F pode ser calculado de forma linear pelo método *SVD* (*Singular Value Decomposition*) [13] utilizando 8 pares de pontos.

4.2. Algoritmo de 7 pontos

Utilizando somente 7 pares de pontos m e m', é possível resolver (9) impondo a restrição de a característica de F ser igual a 2 (o sistema de equações torna-se não linear). Tal como para o algoritmo de 8 pontos, é aconselhado normalizar as coordenadas da imagem antes do cálculo de F [14].

4.3. RANSAC

Algoritmo proposto por *Fischler* e *Bolles* [15] que possibilita o cálculo robusto de F, permitindo distinguir pontos de emparelhamento verdadeiros e falsos, denominados de *inliers* e *outliers*, respectivamente. Assim, partindo de um sub-conjunto de pontos m e m', é calculada uma primeira estimativa de F, e com esta são determinados os *inliers* e *outliers* do restante conjunto de pontos. Quanto menor o sub-conjunto inicial, menor é a probabilidade de este estar contaminado por *outliers*. Este processo, repete-se até se ter atingido um certo grau de satisfação: por exemplo, a probabilidade Γ (usualmente 95%) de se ter escolhido um sub-conjunto de *inliers* após n iterações:

$$\Gamma = 1 - (1 - (1 - \varepsilon)^p)^n\,, \tag{10}$$

sendo ε a fracção dos *outliers* e p o número de pontos contidos no sub-conjunto [16]. Os pontos são considerados *outliers* se estiverem demasiado afastados das linhas epipolares.

4.4. *LMedS*

Difere do algoritmo anterior, no facto de a melhor estimação ser aquela que minimiza a média dos erros:

$$r_i^2 = d^2(m_i', Fm_i) + d^2(m_i, F^T m_i') . \quad (11).$$

Neste algoritmo, a cada correspondência, é atribuído o seguinte peso:

$$w_i = \begin{cases} 1 & r_i^2 \le t^2 \\ 0 & otherwise \end{cases} . \quad (12)$$

A matriz F é então calculada resolvendo o seguinte problema de minimização:

$$\min \sum_i w_i r_i^2 . \quad (13).$$

4.5. *MAPSAC*

Este método, re-define (12), de forma a atribuir uma penalidade aos *outliers* e um peso aos *inliers*, consoante estes se adequam mais ou menos à geometria do sistema:

$$w_i = \begin{cases} r_i^2 & r_i^2 \le t^2 \\ t^2 & otherwise \end{cases} . \quad (14).$$

4.6. Métodos utilizados pelos programas

O programa 1, calcula a geometria epipolar pelo método de *RANSAC*, utilizando o algoritmo de 8 pontos para o cálculo das estimativas de F em cada iteração e termina, onde desta vez utiliza somente os pontos *inliers* encontrados.
Já o programa 3, procede de igual forma, mas fornece a possibilidade de optar entre o método de *RANSAC* ou *LMedS*.
Por seu lado, o programa 2 utiliza o método de *MAPSAC*, sendo as estimativas de F realizadas pelo algoritmo de 7 pontos. No final, após a determinação dos pontos *inliers* e *outliers*, F é calculado pelo algoritmo de 8 pontos, tendo o cuidado de minimizar o ruído introduzido pela utilização de mais de 8 pontos.

5. RECTIFICAÇÃO

Se a geometria epipolar (traduzida pela matriz fundamental) é conhecida, é possível restringir o problema da correspondência de pontos entre duas imagens usando as linhas epipolares. Melhor ainda, se as imagens forem alteradas de forma a colocar as linhas epipolares paralelas ao eixo horizontal da imagem e se forem as mesmas em ambas as imagens - rectificação -, o problema do emparelhamento de pontos torna-se ainda mais fácil de resolver. Somente o programa 5 realiza esta operação.

5.1. *Projective Rectification without Epipolar Geometry (programa 5)*

Rectifica de uma forma projectiva duas imagens estéreo, sem para isso ser necessário a determinação da geometria epipolar, ou mais especificamente, da matriz fundamental.
Ao invés de determinar os pontos epipolares, e com eles determinar duas homografias H e H' que os mapeassem para o infinito, como geralmente fazem os outros algoritmos de rectificação [17], este algoritmo explora o facto de que a matriz fundamental $\overline{F}$ de um par de imagens rectificadas tem uma forma particular e bem conhecida:

$$\overline{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}. \tag{15}$$

Obtidas as homografias de rectificação H e H', é então possível calcular a matriz fundamental *F*:

$$F = H'^{T} \overline{F} H \,. \tag{16}.$$

Combinando (7) e (16), verifica-se que é possível obter H e H' directamente dos pontos de correspondência entre duas imagens [7], mesmo em presença de ruído e de *outliers*, minimizando a função de custo $\mathcal{F}$:

$$\mathcal{F}(H, H') = \sum_{i=1}^{N} [(H' m_i')^T \overline{F} H m_i]^2 \,. \tag{17}.$$

A minimização de (17), é um problema não-linear de minimização de mínimos quadrados, que é resolvida com o algoritmo iterativo de *Levenberg-Marquardt* [18], que necessita de uma boa estimação inicial de H_0 e H_0' para garantir convergência. Tal é obtido minimizando:

$$\mathcal{F}_{\mathcal{R}}(H, H') = \sum_{i=1}^{N} \rho((H' m_i')^T \overline{F} H m_i), \tag{18}$$

em que $\rho(z) = \log(1 + 1/2z)$ - distribuição Lorentziana [18], em imagens de baixa resolução. A partir de H_0 e H_0', filtra-se os *outliers* rejeitando aqueles cujo resíduo r_i é superior a $5.2 \times med_i \left\{ \left| r_i - med_j r_j \right| \right\}$, com:

$$r_i = \left| (H_0' m_i')^T \overline{F} H_0 m_i \right|. \tag{19}.$$

Obtidos H_0 e H_0', itera-se (17) em imagens de resolução cada vez mais elevada, até se ter obtido H e H', a partir das quais se calcula $\overline{F}$ (16).

6. MAPA DE DISPARIDADE

Num sistema estéreo, a informação 3D de uma cena é representada pela disparidade entre duas imagens. O mapa de disparidade, codifica a distância dos objectos em relação à(s) câmara(s), ou seja, pontos muitos distantes têm disparidade zero (geralmente, equivalente ao preto) e pontos muito próximos terão a máxima disparidade (geralmente 255, correspondente ao branco). Em suma, um mapa de disparidade, dá a percepção das descontinuidades em termos de profundidade de uma cena.
Somente os programas 3 e 6, realizam esta operação, utilizando o algoritmo de *Stan Birchfield* [19], com a diferença de que o 6 também retorna um mapa de descontinuidades, definido pelos pixéis que fazem fronteira entre uma mudança de pelo menos dois níveis de disparidade.

6.1. Algoritmo de *Stan Birchfield*

Desenvolvido em 1999, toma como ponto de partida duas imagens rectificadas. Primeiro, o algoritmo realiza um processo de emparelhamento denso, linha a linha na horizontal (daí a necessidade das imagens serem rectificadas); o algoritmo mede a dissimilaridade entre os pontos, atribuindo a cada sequência de correspondências M um custo $\gamma(M)$, que mede o quão improvável é que essa sequência seja uma verdadeira correspondência:

$$\gamma(M) = N_{occ}k_{occ} - N_m k_r + \sum_{i=1}^{N_m} d(m_i, m_i') . \qquad (20).$$

Os parâmetros N_{occ} e N_m, representam o número de oclusões e de emparelhamentos, sendo k_{occ} e k_r as respectivas constantes de penalidade e recompensa, e $d(m_i, m_i')$ a dissimilaridade entre os pixéis m_i e m_i':

$$d(m_i, m_i') = \min\{\overline{d}(m_i, m_i', I, I'), \overline{d}(m_i', m_i, I', I)\} , \qquad (21)$$

em que $\overline{d}$ é uma interpolação linear da intensidade da imagem entre dois pixéis [20].
Partindo do pressuposto de que descontinuidades são acompanhadas de variações na intensidade da imagem, são definidas restrições a cada sequência de correspondências M, tais como impor um limite máximo ao valor de d. A óptima sequência de emparelhamento é calculada utilizando uma técnica de programação dinâmica.
Como a intensidade entre linhas de uma imagem não são independentes, é realizada uma análise coluna a coluna no mapa de disparidades. Primeiro, a disparidade de um pixel, cujos vizinhos verticais tenham disparidade igual mas diferente dele próprio, torna-se igual à dos seus vizinhos. A seguir, para ambas as direcções horizontal e vertical, utiliza-se o gradiente na direcção a estudar da imagem original, para propagar regiões de confiança no mapa de disparidade. A confiança, para cada pixel, define-se como o número de pixéis contíguos na direcção a estudar com disparidade igual à sua. Depois, o mapa de disparidade é filtrado com um filtro de média, nas duas direcções sequencialmente, por

forma a preservar os vértices dos objectos.

7. RESULTADOS

Os resultados experimentais obtidos são apresentados neste artigo segundo a sequência descrita na Figura 1. Foram obtidos utilizando imagens, de dimensões 540×612, de objectos reais com diferentes níveis de complexidade, capturadas com uma câmara digital convencional.

7.1. Extracção de pontos fortes

Na Tabela 1, e nas Figura 2 a Figura 4, estão indicados os resultados obtidos pelos programas 1 a 4, na extracção de pontos fortes nas várias imagens de teste consideradas.

IMAGEM	***Programa 1***	***Programa 2***	***Programa 3***	***Programa 4***
Tlm_1	36	42	30	30
Rato_1	38	62	30	30
Pc_1	21	28	30	30

Tabela 1. Número de pontos fortes obtidos na primeira imagem de cada par de imagens de teste; os programas 3 e 4, têm um parâmetro de entrada que define o número de pontos que se pretende obter.

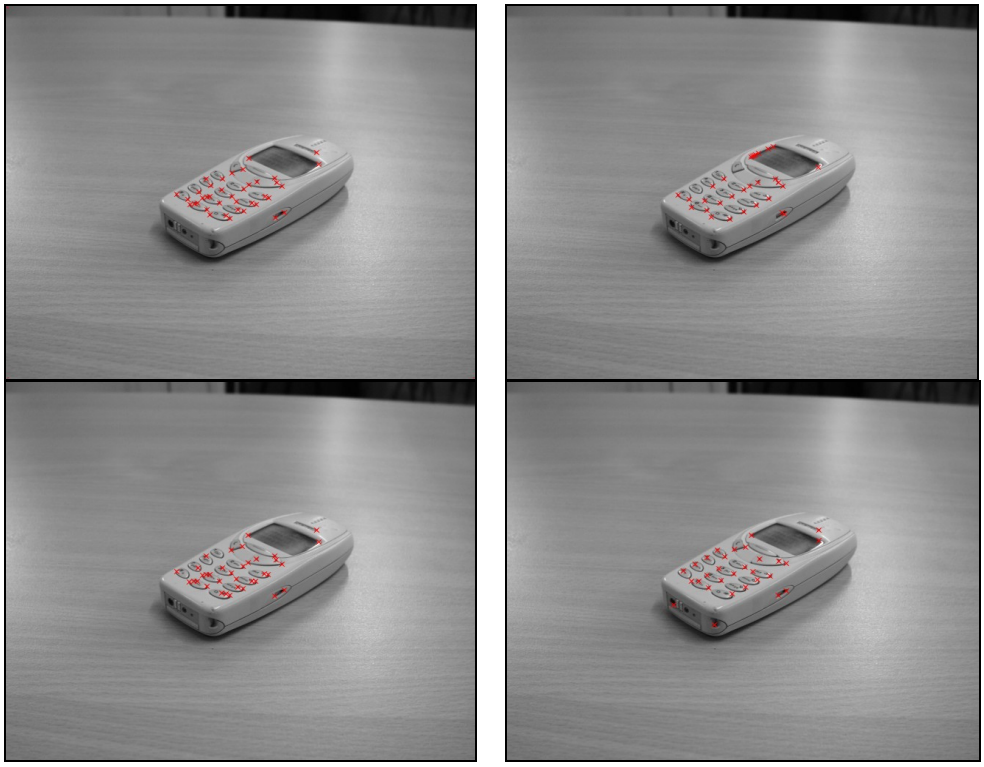

Figura 2. As cruzes vermelhas assinalam os pontos fortes encontrados na imagem *Tlm_1* (da direita para esquerda e de cima para baixo, são os resultados dos programas 1 a 4).

Como se pode observar na Tabela 1 e nas Figura 2 a Figura 4, o programa 2 (*Torr's Matlab Toolkit*) é o que apresentam piores resultados. Todos os outros são bastante semelhantes. É importante reparar que os programas 3 (*OpenCV*) e 4 (*KLT*), limitam a distância mínima entre pontos fortes, o que pode ser relevante numa escolha criteriosa de pontos em zonas da imagem com variações de gradiente mais acentuadas.

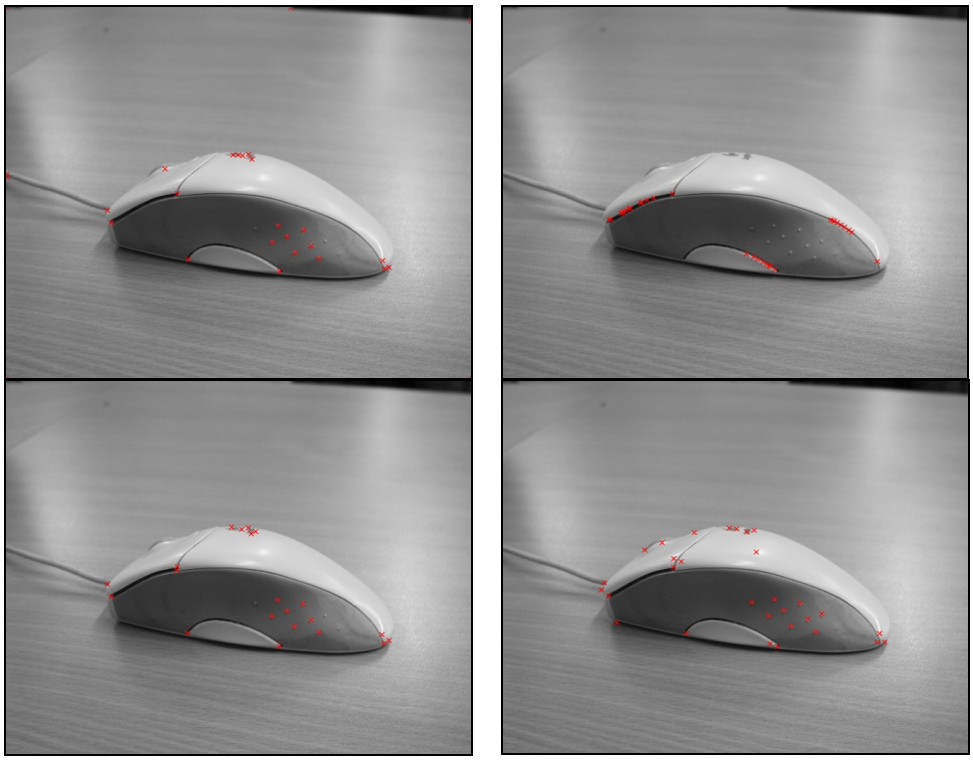

Figura 3. As cruzes vermelhas assinalam os pontos fortes encontrados na imagem *Rato_1* (da direita para esquerda e de cima para baixo, são os resultados dos programas 1 a 4).

7.2. Emparelhamento de pontos fortes

Dados os resultados obtidos na sub-secção anterior, e de forma a garantir a comparabilidade dos resultados mais adequada, nesta fase são utilizados os pontos fortes obtidos na primeira imagem pelo programa 4 (*KLT*). Os resultados obtidos, são indicados na Tabela 2 e nas Figura 5 a Figura 7, tendo sido usado 30 pontos fortes na primeira imagem de cada caso.

Com os resultados obtidos e apresentados nas Figura 5 a Figura 7 e na Tabela 2, observa-se que o emparelhamento de pontos fortes é uma fase crítica em Visão 3D. Em todos programas, se verificam erros na correlação, sendo o programa 2 (*Torr's Matlab Toolkit*) o que apresenta os piores resultados. A pior imagem, é a *Rato* pois o objecto apresenta contornos curvilíneos e sem fortes contrastes.

7.3. Cálculo da geometria epipolar

Dados os resultados obtidos na sub-secção anterior, e mais uma vez de forma a garantir a comparabilidade dos resultados mais adequada, nesta fase são utilizados os pontos de emparelhamento obtidos pelo programa 3 (*OpenCV*). A Tabela 3 e a Figura 8, apresentam os resultados obtidos.

IMAGENS	***Programa 1***	***Programa 2***	***Programa 3***	***Programa 4***
Tlm	30	14	21	27
Rato	23	15	10	24
Pc	28	16	13	22

Tabela 2. Número de emparelhamentos obtidos com sucesso entre pontos de cada par de imagens.

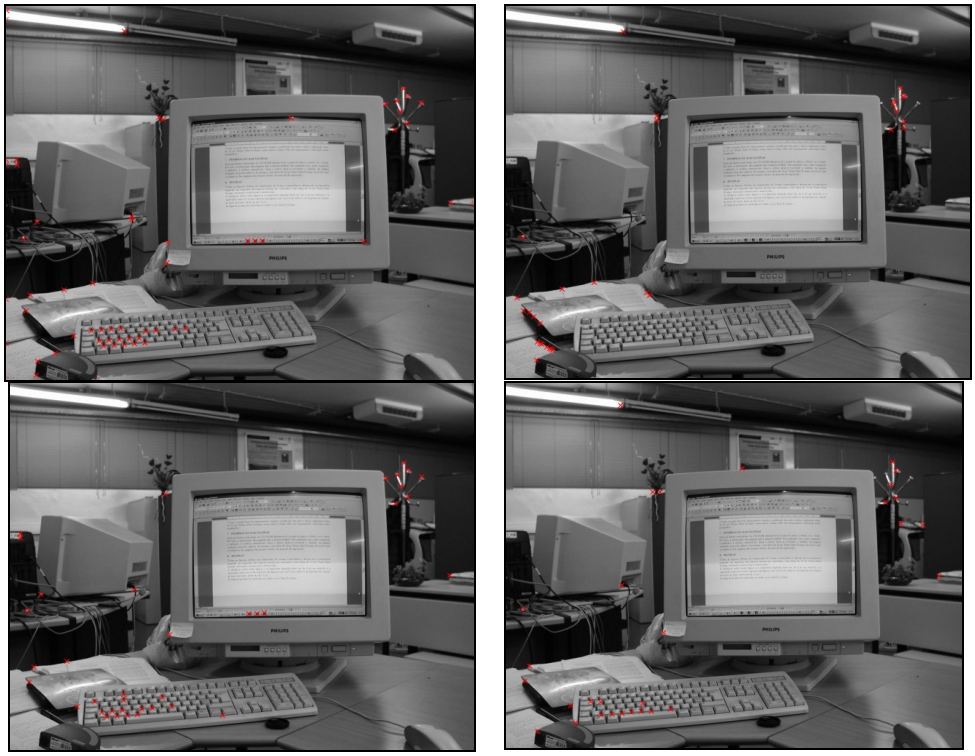

Figura 4. As cruzes vermelhas assinalam os pontos fortes encontrados na imagem *Pc* (da direita para esquerda e de cima para baixo, são os resultados dos programas 1 a 4).

IMAGEN	***Prog. 1 Ransac***	***Prog. 1 LmedS***	***Programa 2***	***Programa 3***
Tlm	26	24	11	-
Rato	18	16	13	-
Pc	22	18	11	-

Tabela 3. Número de *inliers* em cada par de imagens, após o cálculo da geometria epipolar.

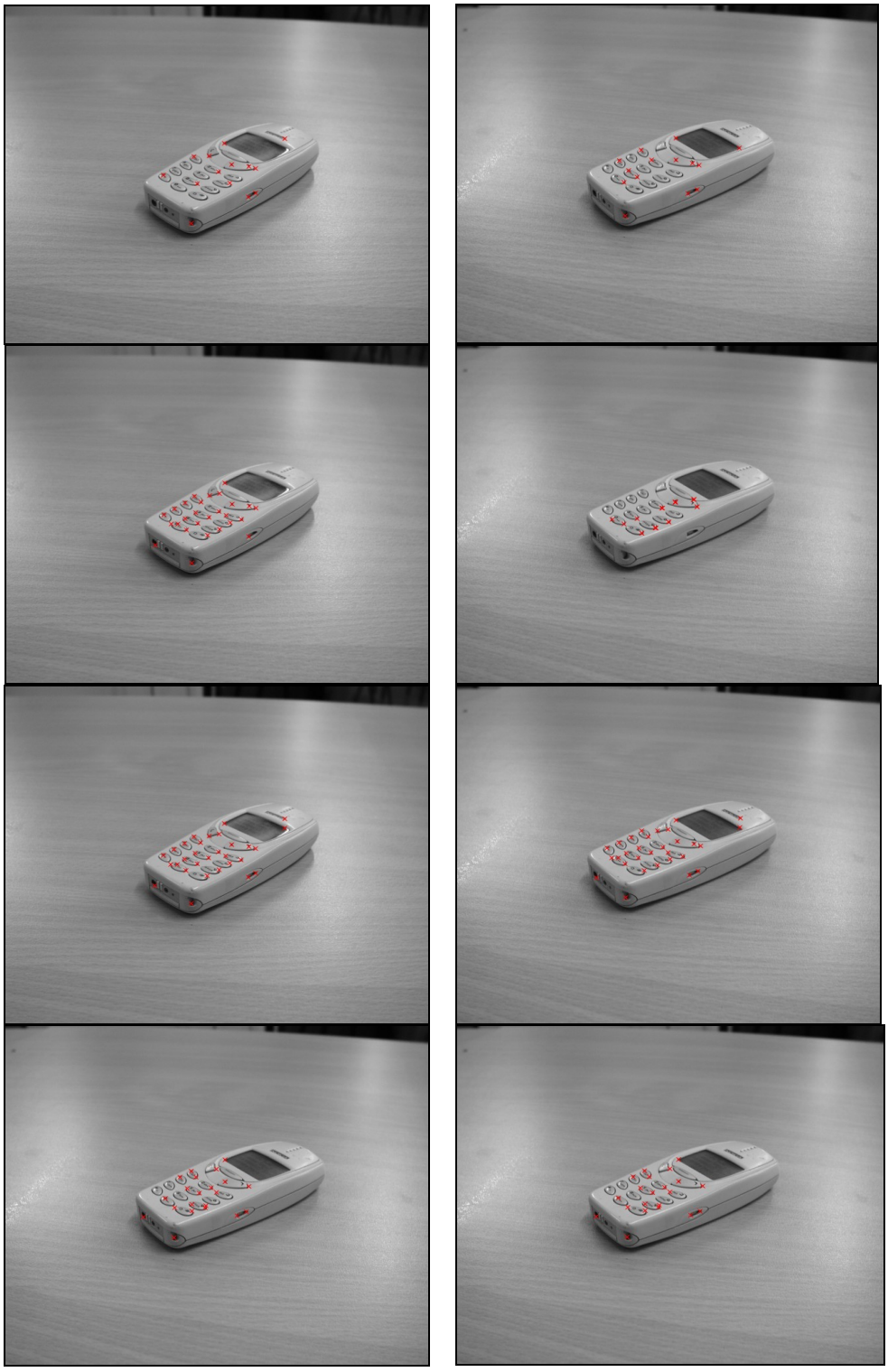

Figura 5. À direita a imagem *Tlm_1*, e à esquerda a imagem *Tlm_2*. As cruzes vermelhas assinalam os pontos de emparelhamento (de cima para baixo, são os resultados dos programas 1 a 4).

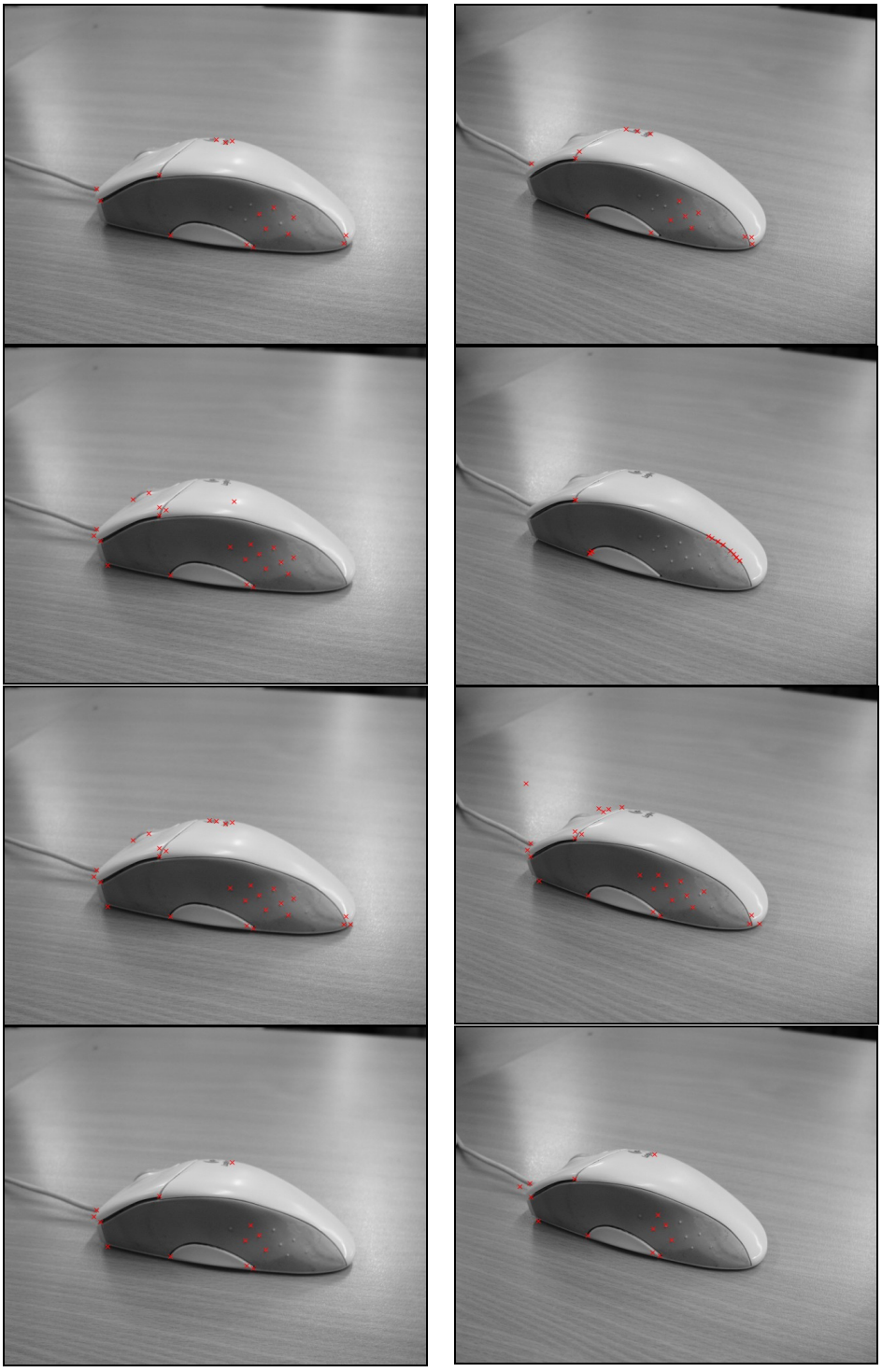

Figura 6. À direita a imagem *Rato_1* e à esquerda a imagem *Rato_2*. As cruzes vermelhas assinalam os pontos de emparelhamento (de cima para baixo, são os resultados dos programas 1 a 4).

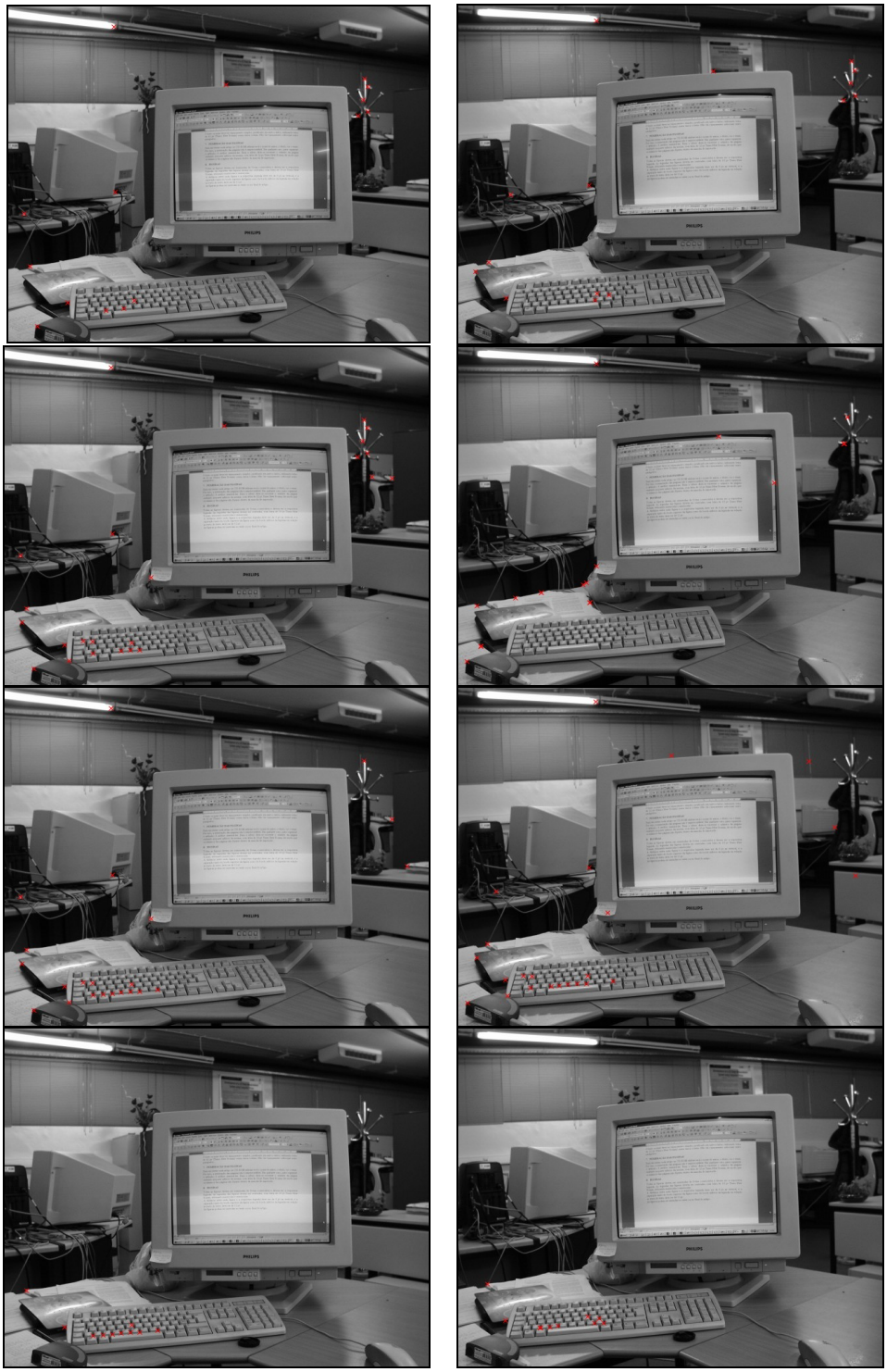

Figura 7. À direita a imagem *Pc_1* e à esquerda a imagem *Pc_2*. As cruzes vermelhas assinalam os pontos de emparelhamento (de cima para baixo, são os resultados dos programas 1 a 4).

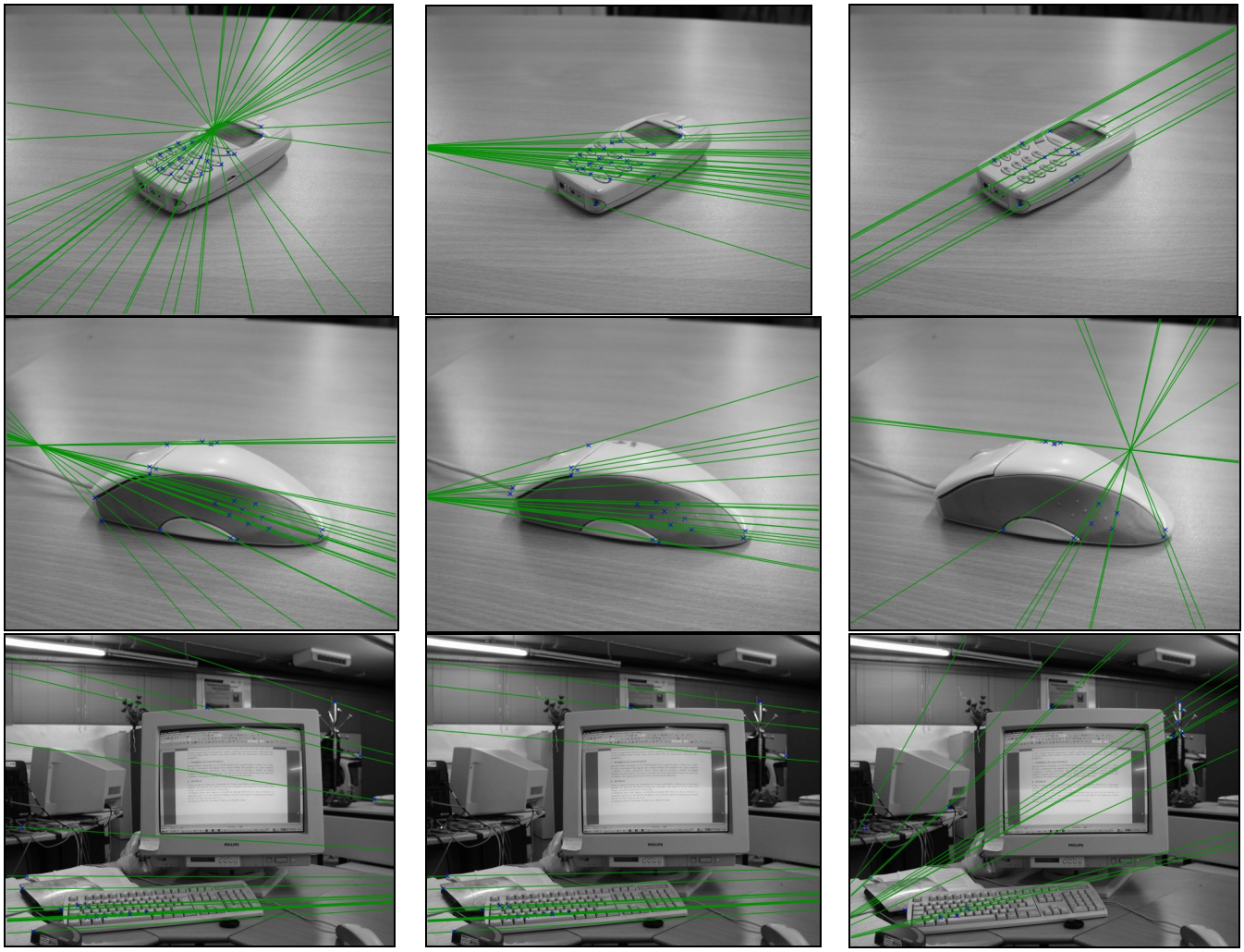

Figura 8. Linhas epipolares (a verde) e *inliers* (a azul) obtidos na primeira imagem de cada par de imagens de teste; da esquerda para a direita, resultado do programa 1 (método de *Ransac*), resultado do programa 1 (método de *LMedS*) e resultado do programa 2.

A Tabela 3 e a Figura 8, permitem algumas conclusões: Como os maus resultados obtidos pelo programa 2 (*Torr's Matlab Toolkit*) nas sub-secções 7.1 e 7.2 faziam prever, não foi possível obter a geometria epipolar utilizando os seus métodos, visto o número de *outliers* ser substancialmente superior aos de *inliers*. Dos restantes programas, o que aparenta melhores resultados é o programa 3 (*OpenCV*), embora se note a necessidade de obter pontos de emparelhamento mais adequados no sentido de se conseguir melhores resultados no cálculo da geometria epipolar.

7.4. Rectificação

Como o programa 5 (*Projective Rectification without Epipolar Geometry*) necessita, para além de um par de imagens estéreo, de alguns pontos de emparelhamento, foram utilizados os

resultados da sub-secção 7.2. Os resultados obtidos, estão apresentados na Figura 9.

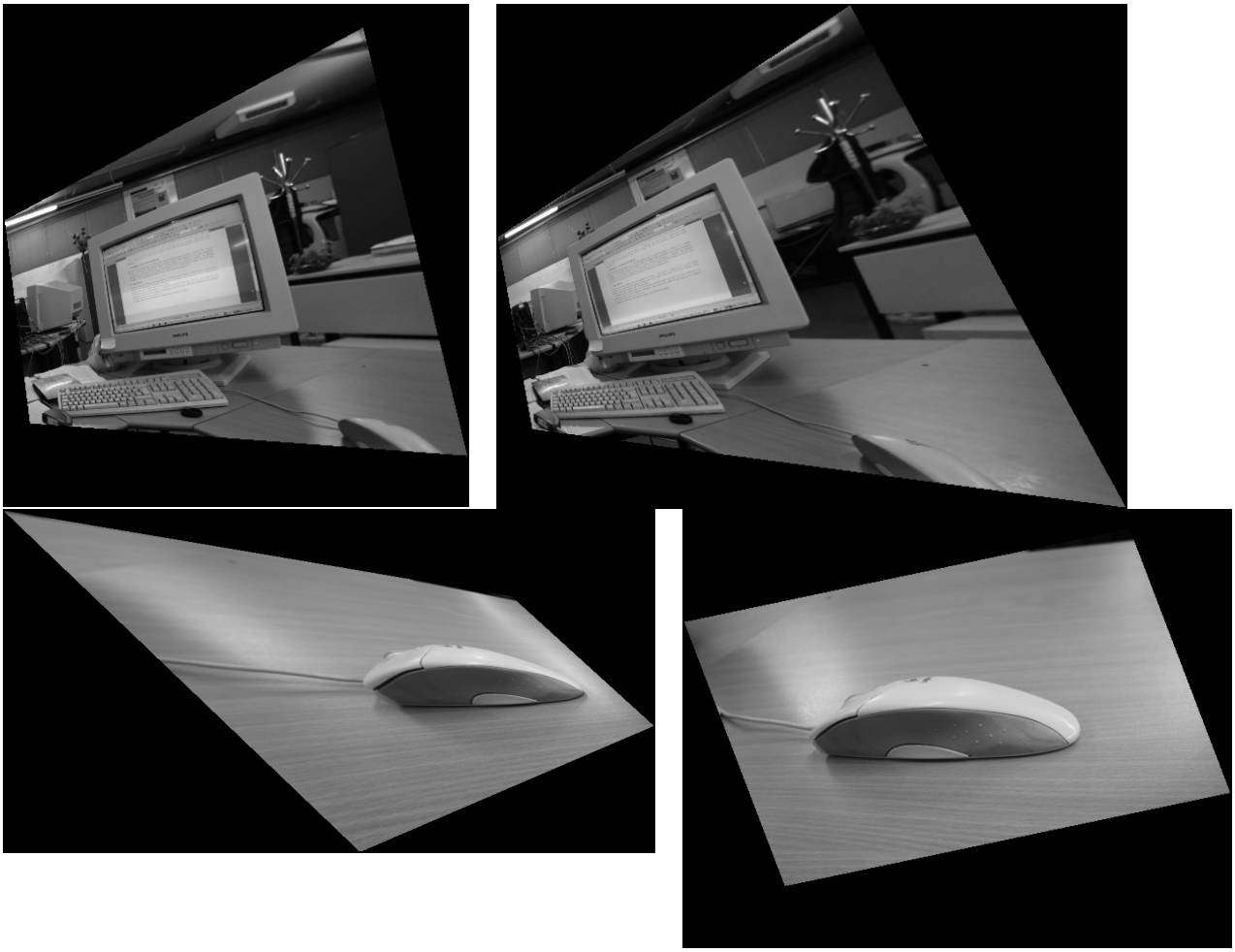

Figura 9. Resultado da rectificação das imagens de teste pelo programa 5.

Nota: Não foi possível proceder à rectificação do par de imagens *Rato_1* e *Rato_2*, visto que as matrizes de homografia H e H' (16) calculadas pelo programa 5 eram nulas.

7.5. Emparelhamento denso

Esta etapa foi testada com imagens, previamente rectificadas e indicadas pelo programa 6 (*Depth Discontinuities by Pixel-to-Pixel Stereo*), apresentadas na Figura 10. Os resultados obtidos, estão apresentados nas Figura 11 e Figura 12

8. CONCLUSÕES E PERSPECTIVAS DE TRABALHO FUTURO

Neste artigo, foram apresentadas e analisadas várias funções desenvolvidas no âmbito da reconstrução 3D de objectos, disponíveis em domínio público.

Figura 10. Imagens originais utilizadas pelos programas 3 e 6 para realizar o emparelhamento denso.

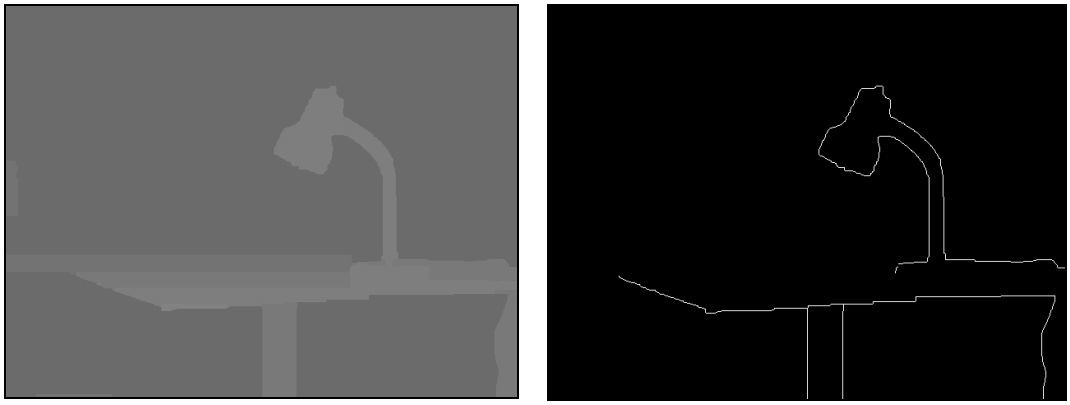

Figura 11. Mapas de disparidade (esquerda) e descontinuidade (direita) obtidos pelo programa 6.

Figura 12. Mapa de disparidade obtido pelo programa 3.

Ao longo deste trabalho, foi desenvolvida uma aplicação computacional, com uma interface gráfica adequada, para permitir a aplicabilidade e a comparação das funções consideradas.
Em resumo, pode-se afirmar que as funções analisadas obtêm bons resultados experimentais quando aplicadas em objectos que apresentam características fortes. Pois os resultados de

menor qualidade, estão relacionados com a determinação dos pontos relevantes nas imagens a considerar e do emparelhamento entre estes. Esta debilidade, dos programas analisados, é tanto maior, quanto menor as variações da forma dos objectos em causa. Assim, como as funções a seguir utilizadas para obter a reconstrução 3D consideram estes pontos relevantes, os resultados obtidos pelas mesmas estão fortemente relacionados com a qualidade destes.
As próximas etapas deste projecto, irão concentrar-se no sentido de melhorar os resultados obtidos pelas metodologias apresentadas quando os objectos a reconstruir apresentam formas contínuas e suaves. Posteriormente, as metodologias desenvolvidas serão aplicadas na reconstrução e caracterização de formas 3D anatómicas exteriores.

REFERÊNCIAS

[1] P. Kovesi, *MATLAB Functions for Computer Vision and Image Analysis*, 2004, http://www.csse.uwa.edu.au/~pk/Research/MatlabFns.

[2] P. Torr, *A Structure and Motion Toolkit in Matlab*, 2002, http://wwwcms.brookes.ac.uk/~philiptorr/Beta/torrsam.zip.

[3] *OpenCV - Open Computer Vision Library*, beta 4, 2004, http://sourceforge.net/projects/opencvlibrary.

[4] S. Birchfield, *KLT: An Implementation of the Kanade-Lucas-Tomasi Feature Tracker*, 2004, http://www.ces.clemson.edu/~stb/klt/index.html.

[5] B. D. Lucas e T. Kanade, *An Iterative Image Registration Technique with an Application to Stereo Vision*, International Joint Conference on Artificial Intelligence, pp. 674-679, (1981).

[6] C. Tomasi e T. Kanade, *Detection and Tracking of Point Features*, Carnegie Mellon University Technical Report CMU-CS-91-132, (1991).

[7] F. Isgrò e E. Trucco, *Projective rectification without epipolar geometry*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR99, Fort Collins (Colorado), vol. I, ISBN 0-7695-0149-4, pp. 94-99, (1999).

[8] S. Birchfield, *Depth Discontinuities by Pixel-to-Pixel Stereo*, 1999, http://vision.stanford.edu/~birch/p2p/.

[9] C. Harris e M. Stephens, *A combined corner and edge detector*, Forth Alvey Vision Conference, pp. 147-151, (1988).

[10] A. Noble, *Description of image surfaces*, PhD thesis, Department of Engineering Science, Oxford University, pp. 45, (1989).

[11] J. Bouguet, *Pyramidal Implementation of the Lucas Kanade Feature Tracker*, Microprocessor Research Labs, Intel Corporation, (1999).

[12] O. Faugeras, Q.-T. Luong, e T. Papadopoulo, *The Geometry of Multiple Images*, MIT Press, (2001).

[13] G. Golub e C. V. Loan, *Matrix Computations*, John Hopkins University Press, (1983).

[14] R. Hartley, *In defense of the eight-point algorithm*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 19, No. 6, pp. 580-593, (1997).

[15] M. Fischler e R. Bolles, *RANdom SAmpling Consensus: a paradigm for model fitting with application to image analysis and automated cartography'*, Commun. Assoc.

Comp. Mach., Vol. 24, No. 6, pp. 381-395, (1981).
[16] P. Rousseeuw, *Robust Regression and Outlier Detection*, Wiley, New York, (1987).
[17] R. I. Hartley, *Theory and Practice of projective Rectification*, International Journal of Computer Vision, (1998).
[18] W. H. Press, S. A. Teukolsky, W. T. Vetterling, e B. P. Flannery, *Numerical recipes in C: the art of scientific computing*, Cambridge University Press, 2nd edition, (1992).
[19] S. Birchfield e C. Tomasi, *Depth Discontinuities by Pixel-to-Pixel Stereo*, International Journal of Computer Vision, vol. 35, no. 3, pp. 269-293, (1999).
[20] S. Birchfield e C. Tomasi, *A Pixel Dissimilarity Measure that is Insensitive To Image Sampling*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, No. 4, pp. 401-406, (1998).