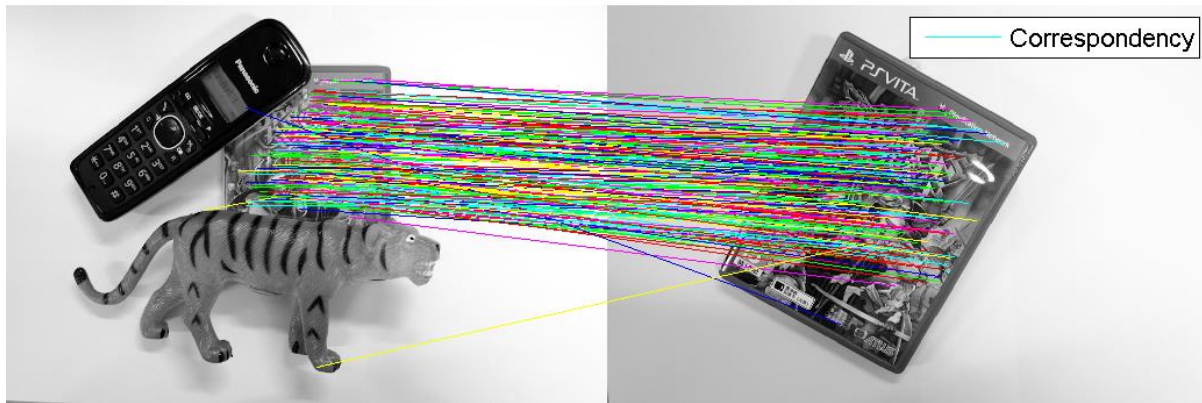


Object recognition using SIFT

Corresponding feature between target & feature



報告內容：

1. SIFT 簡介
2. 方法
 - SIFT: Scale-Invariant Feature Transformation
 - K-NN: K nearest neighbors
 - RANSAC: RANdom Sample Consensus
3. 結果
4. 討論
5. 使用方式

SIFT 簡介：

尺度不變特徵轉換(Scale-invariant feature transform 或 SIFT)是一種電腦視覺的演算法用來偵測與描述影像中的局部性特徵，它在空間尺度中尋找極值點，並提取出其位置、尺度、旋轉不變數，此演算法由 David Lowe 在 1999 年所發表，2004 年完善總結。其應用範圍包含物體辨識、機器人地圖感知與導航、影像縫合、3D 模型建立、手勢辨識、影像追蹤和動作比對。

此演算法有其專利，專利擁有者為 英屬哥倫比亞大學。

SIFT: Scale-Invariant Feature Transformation:

礙於報告篇幅，本節不詳細描述演算法過程，只稍加敘述其概念。SIFT 演算法中，利用 Laplacian filter 在 scale space 中定位出影像中的 local extrema，在此稱為 keypoint，然後利用對 keypoint 周圍各 pixel 在 x 和 y 方向取差分，取得 pixel 的 gradient 方向，並統計所有 pixel 的貢獻以訂定該 keypoint 的主方向。最後，統計某 keypoint 周圍的 pixel gradient” 相對” 於主方向的方向，給予 Descriptor。由於 SIFT descriptor 為一個 128 維的向量，所以能在海量的 descriptors 中保有一定的唯一性。



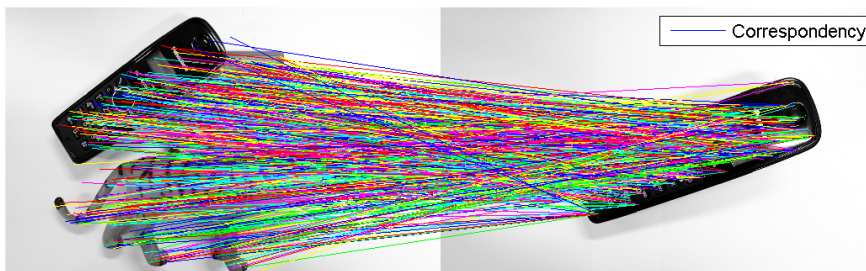
FIGURE.1

SIFT 在影像上所找到的 keypoint

由於 SIFT descriptor 為一個 128 維的向量，所以能在海量的 descriptors 中保有一定的唯一性。

keypoint 的偵測如 Figure.1(a)所示，但如果單純利用 SIFT descriptors 作物體辨認的話存在相當大 ambiguous，那就需要利用 K-NN 的觀念去除掉特徵不相似的點。

Corresponding feature between target & feature_{w/o} ratio threshold

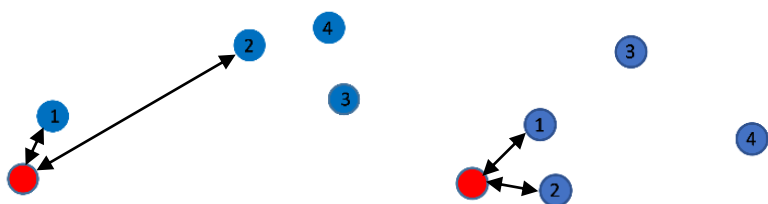


a b

FIGURE.2

(a) Target image (b) Object image 1st nearest neighbor 的 keypoint 的對應關係。

K-NN: K nearest neighbors:



a b

FIGURE.3

(a) Target descriptor 和配對的 1st、2nd nearest neighbor 距離相差很遠，屬於 1st nearest neighbor 類的機率很高

當有一個新的 target descriptor，要與 training data 中 object descriptors 作配對時，需要和 K 個作比對。當 K=1，就是取其最近距離 object descriptor 的作為配對；在作業中，取 K=4 並記錄此 4 個 object descriptors 與 target descriptor 的距離。

在上一節提過，SIFT descriptor 仍存有一定的 ambiguity，為了避免選取到 outlier，我們需要確定配對的準確性，此時用 target descriptor 與 1st nearest neighbor 的距離和 target descriptor 與 2nd nearest neighbor 的距離，兩個距離之間的比值作為第一階段判斷是否為 outlier 的依據，以降低下一步 RANSAC 的運算量。

Corresponding feature between target & feature

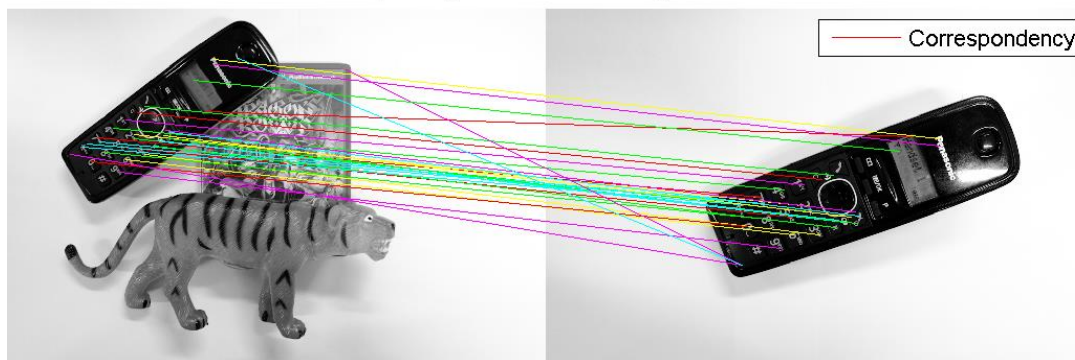


FIGURE.3

定義當 target 與 1st、2nd nearest neighbor 兩者之間筆直大於 0.7 時，認定為 outlier 並捨棄配對點

RANSAC: RANdOm Sample Consensus:

當資料點含有定數量的 inlier 能被某種數學模型描述，但同時資料點也含有 outlier 無法使用最小平方法時，就可以利用 RANSAC 來排除掉 outlier 建立模型。

此次 project 需要計算 Projective matrix:

$$\begin{pmatrix} wx \\ wy \\ w \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}$$

$$x(XA_{31} + YA_{32} + A_{33}) = (XA_{11} + YA_{12} + A_{13})$$

$$y(XA_{31} + YA_{32} + A_{33}) = (XA_{21} + YA_{22} + A_{23})$$

利用四組”正確”的對應點，我就能有四組 equation，幫助我們求解 $A_{11} \sim A_{33}$ 。

RANSAC 演算法為(4-NN 的狀況下):

(1) 設定總共要跑幾個 round: z ，不同的 round 會挑不同 4 個 obj keypoint

Round =1~ z

(2) 在每個 round 有四個 random number: n_1, n_2, n_3, n_4

分別代表四個 obj keypoints 的 index

(3) 分別存取四個 obj keypoints 的 nearest neighbors(共 $4*4*4*4$ 種可能)

(4) 每次都要算 Homography matrix，並計算 inlier 數

end

選取 inlier 數目最多的那組作為正確的 Homography matrix，將 object keypoint 座標轉換到 target image 上。

Object feature points transformation



FIGURE.3

將(a)object keypoint 座標轉換到(b)target image 上

結果:

Datasets 是由林奕成教授提供，並作為課堂 project 使用。

Data:



Object image:

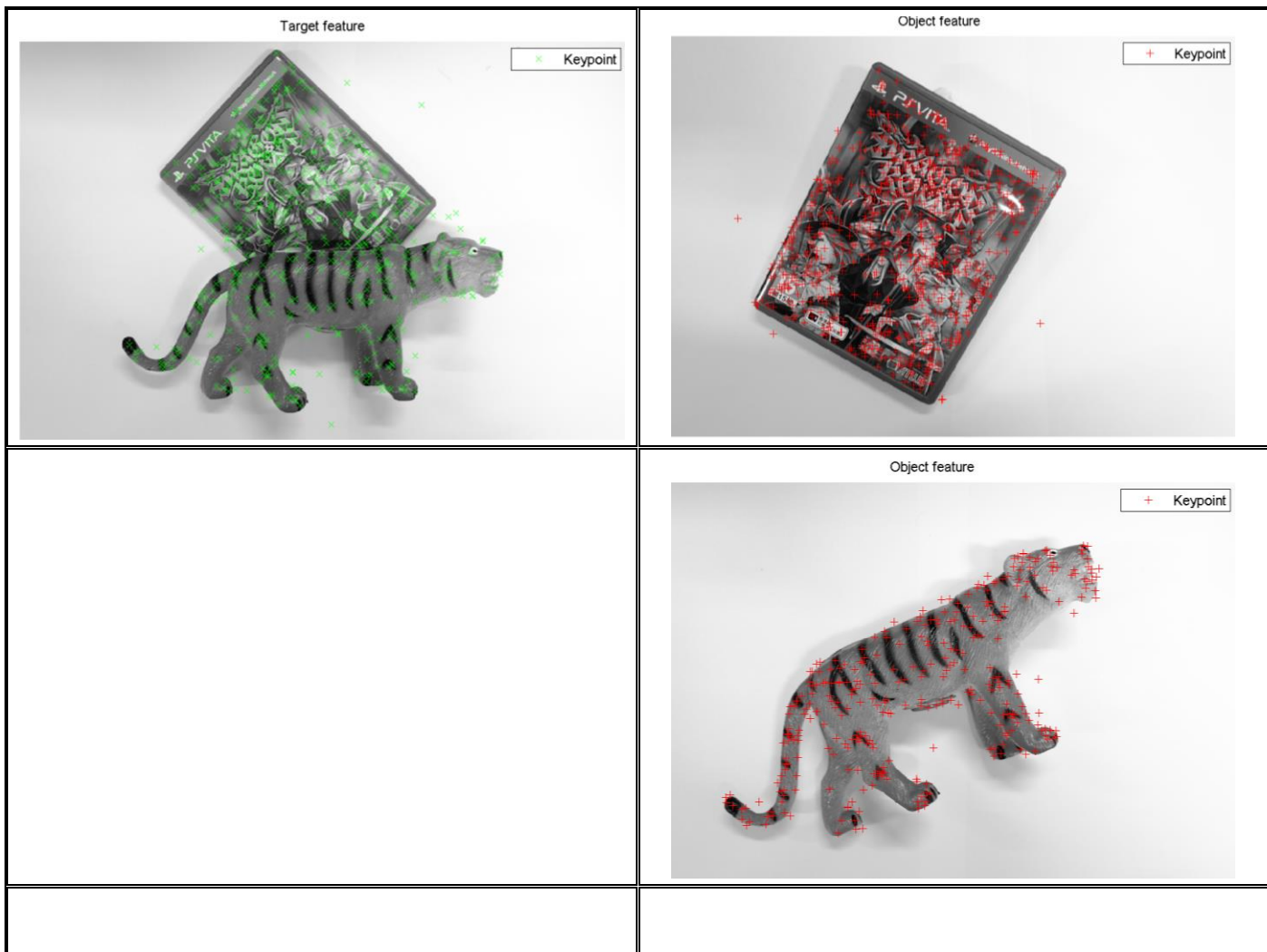


Target image:



I. Feature Detection

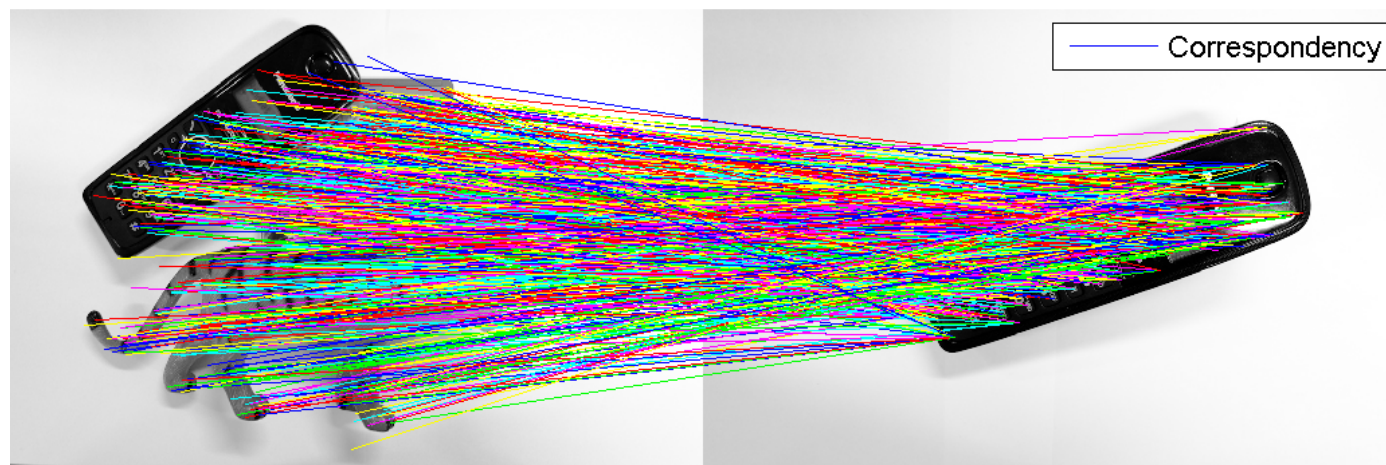
Target Image	Object Image
<p data-bbox="395 1355 496 1377">Target feature</p>  <p data-bbox="651 1391 767 1413">x Keypoint</p>	<p data-bbox="1098 1355 1198 1377">Object feature</p>  <p data-bbox="1353 1391 1469 1413">+ Keypoint</p>



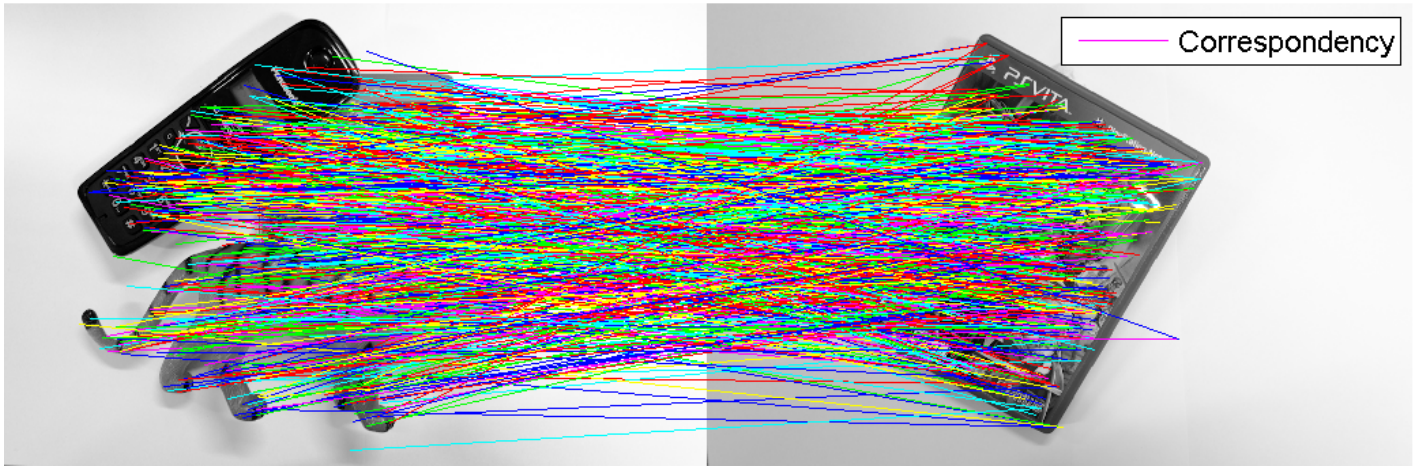
II. Feature Correspondence

Target feature 與 Object feature 對應關係(無 Ratio 限制)

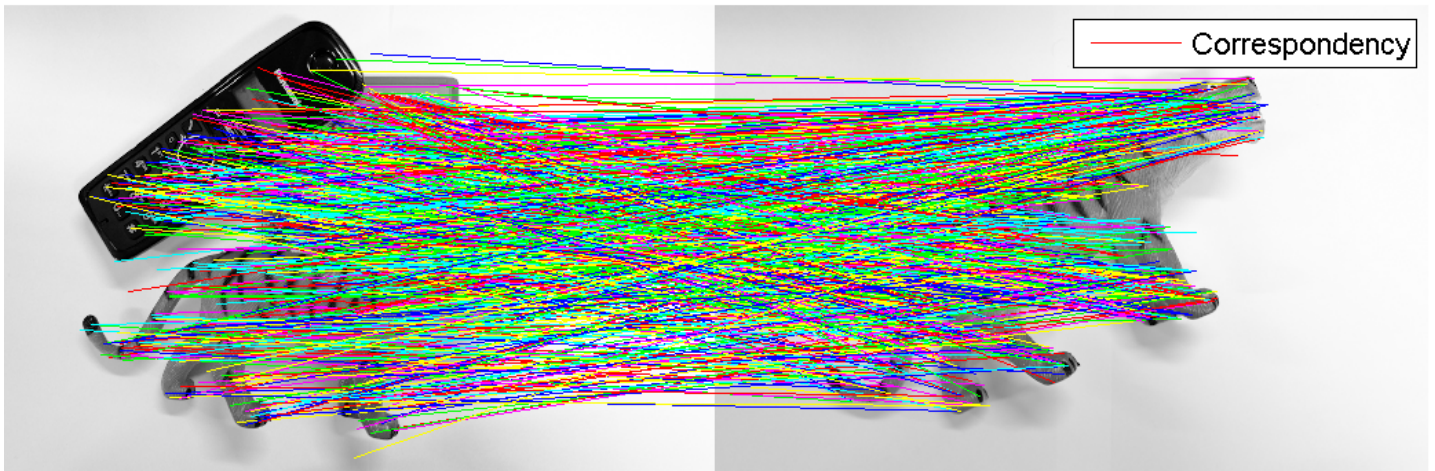
Corresponding feature between target & feature_{w/o} ratio threshold



Corresponding feature between target & feature_{w/o} ratio threshold

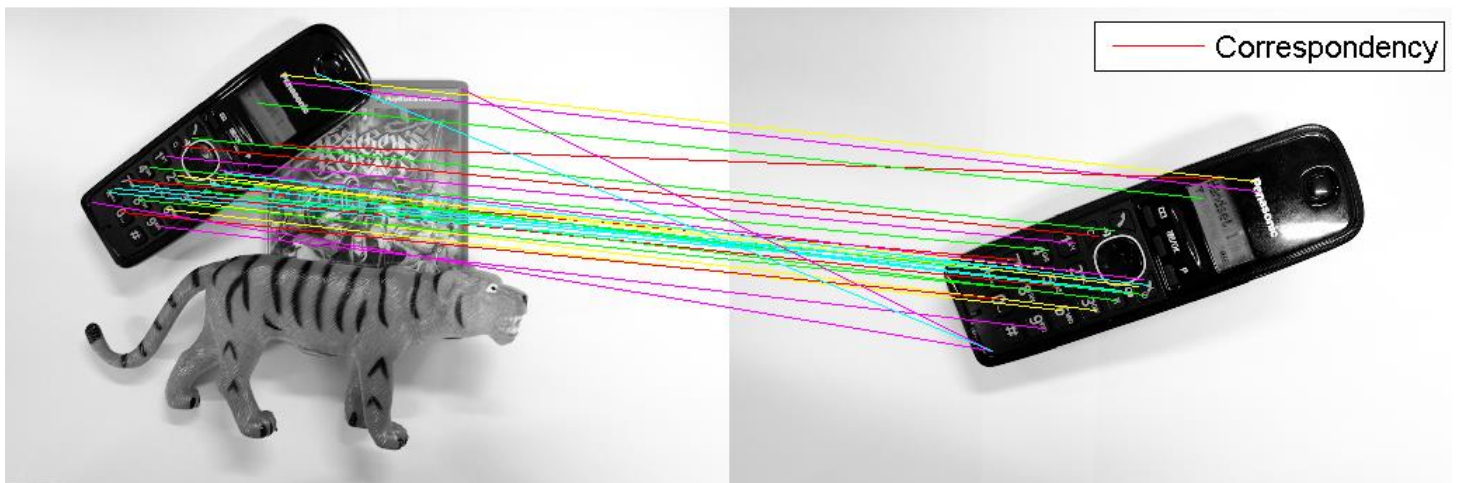


Corresponding feature between target & feature_{w/o} ratio threshold

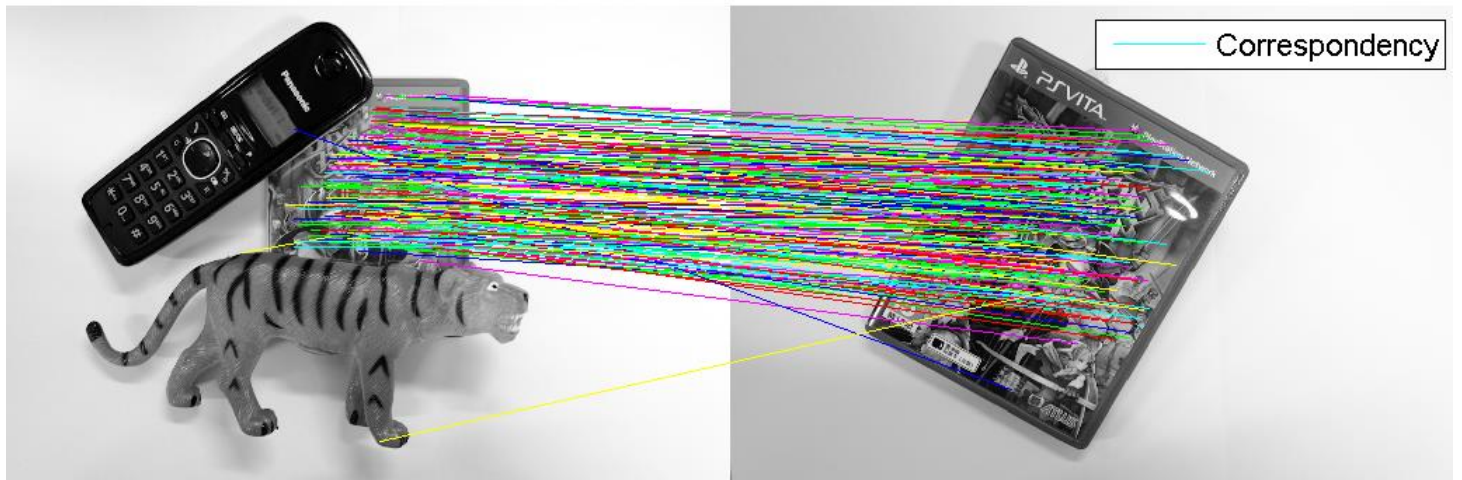


Target feature 與 Object feature 對應關係(Ratio 限制為 0.7)

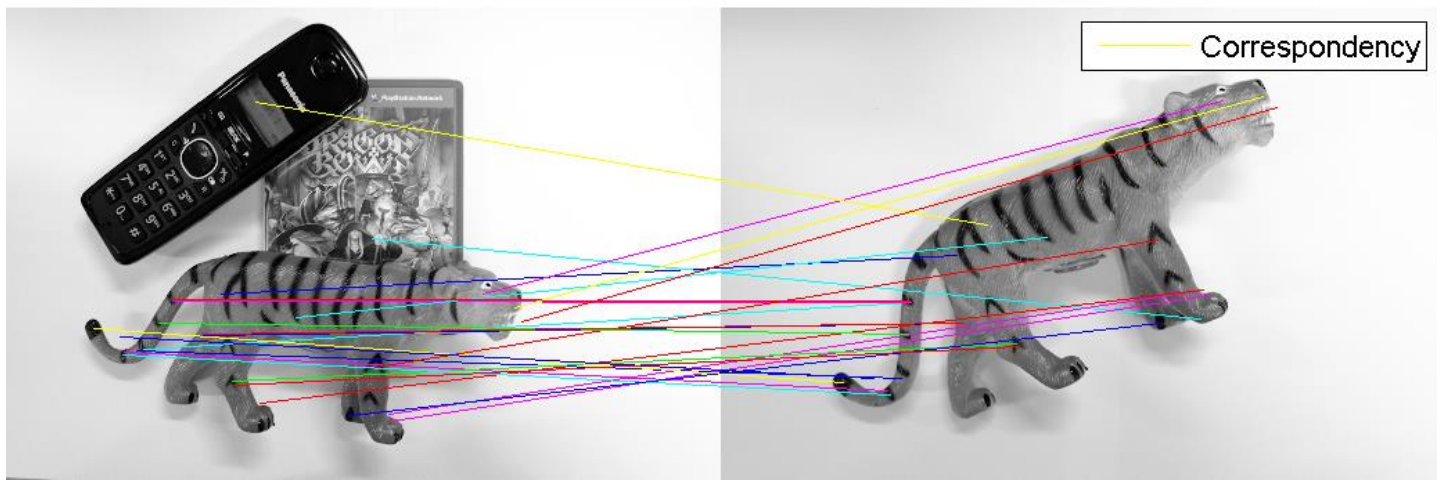
Corresponding feature between target & feature



Corresponding feature between target & feature



Corresponding feature between target & feature



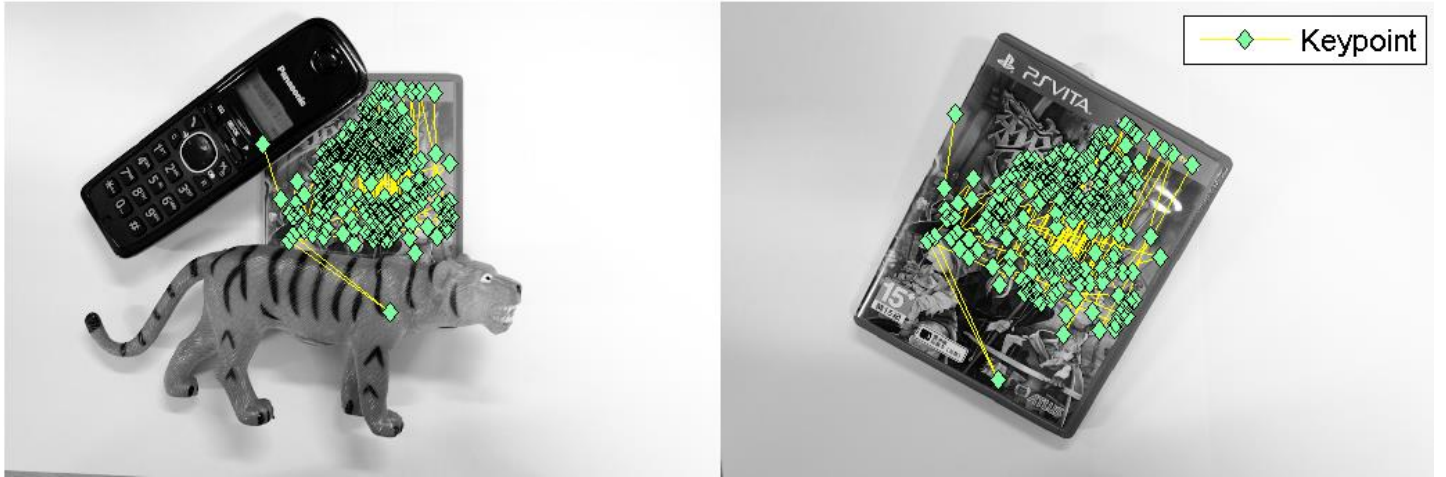
III. RANSAC & Projective Mapping

Target feature 與 Object feature 對應關係(Ratio 限制為 0.7)

Object feature points transformation



Object feature points transformation



Object feature points transformation

