

# Line-Based Affine Invariant Object Location Using Transformation Space Decomposition

Richard Yang and Yongsheng Gao

Computer Vision and Image Processing Lab, School of Engineering,  
Griffith University, Brisbane, Australia

richard.yang@student.griffith.edu.au, yongsheng.gao@griffith.edu.au

## Abstract

*This paper presents a novel line-based affine invariant object location methodology. Our algorithm employs a new line-based transformation space decomposition technique to exploit intrinsic structural information provided by line features. Furthermore, we propose a new line-based distance transform to integrate with our algorithm to provide efficient transformation cell evaluation and subdivision in a coarse to fine manner. The algorithm is able to rapidly accelerate the searching process while maintaining high discriminative power and minimal storage requirement. The efficiency and discriminative power of this methodology are demonstrated using real-world examples with promising results.*

## 1. Introduction

One of the most difficult and challenging tasks for a vision system is to detect objects from different viewing angles. Local information is used in recognizing affine invariant planar objects based on boundary-based Fourier descriptors [1, 3], interest-point detector [7] and wavelet-based descriptors [4]. Rucklidge [5] used point based edge map as local information to efficiently locate affine transformations of objects using transformation space subdivision. However, point based object contour and edge map methods are sensitive to noise and clutter. In addition, point based transformation space pruning only utilizes the spatial information of an edge map without considering the inherent local structural characteristics. Furthermore, the evaluate and subdivide method in [5] explicitly considers every pixel on an edge in evaluating transformation cells, leading to ineffective pruning and longer calculation time in dense images. It is shown that lines offer greater discriminative power and achieved improved recognition accuracy, superior storage requirement and quicker recognition speed

when compared to point based approach [6]. Line feature represents an intuitive progress from point based edge map representation due to line feature's efficient storage capability and inherent structural information. However, method in [6] assumed that the location of objects are known and consequently can not be applied to the task of affine object location.

This paper proposes a novel line-based affine invariant object location methodology. This approach reduces storage requirement by grouping pixels to line segments. The proposed methodology is also surprisingly efficient in both reducing search space and search complexity while maintaining high detection accuracy. We also develop a line based distance transform to facilitate efficient decomposition of transformation cells.

## 2. Line Based Distance Transform

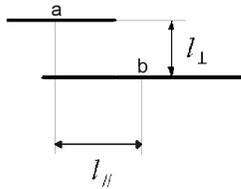
Given a set of lines  $A = \{\bar{a}_1, \bar{a}_2, \dots, \bar{a}_n\}$  representing a target image to be searched, each line segment  $\bar{a}_i$  is represented by its mid-point and angle:

$$\bar{a}_i = [x_i \quad y_i \quad \theta_i]^T \quad (1)$$

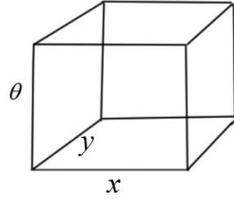
We define the line based distance transform of a line set as a three dimensional cube with axes  $x, y$  and  $\theta$ , as illustrated in Fig. 2.

$$\Delta^l[x, y, \theta] = \min_{l \in l} d((x, y, \theta), \bar{a}) \quad (2)$$

$x$  and  $y$  are bounded by the image dimension and  $\theta \in [0, \pi]$ . The value of a point  $(x_i, y_j, \theta_k)$  in the cube represents the minimum line segment based distance from the line with parameter  $x_i, y_j, \theta_k$  to any line segment in the line set.



**Figure 1. Perpendicular and parallel distance.** *a* and *b* are mid-points



**Figure 2. Line based distance transform.** *x*, *y* denotes mid-point, *theta* denotes line angle

The line segment based distance consists of three aspects of differences between line segments: perpendicular distance ( $d_{\perp}$ ), parallel distance ( $d_{\parallel}$ ) and orientation distance ( $d_{\theta}$ ). Similar to [6], in this paper they are defined as:

$$d_{\parallel}(\bar{m}_i, \bar{t}_j) = l_{\parallel} \quad (3)$$

$$d_{\perp}(\bar{m}_i, \bar{t}_j) = l_{\perp} \quad (4)$$

$$d_{\theta}(\bar{m}_i, \bar{t}_j) = f(\theta(\bar{m}_i, \bar{t}_j)) \quad (5)$$

The distance between two line segments is the Euclidean distance of all three distances combined:

$$d(\bar{m}_i, \bar{t}_j) = \sqrt{d_{\parallel}^2(\bar{m}_i, \bar{t}_j) + d_{\perp}^2(\bar{m}_i, \bar{t}_j) + d_{\theta}^2(\bar{m}_i, \bar{t}_j)} \quad (6)$$

Orientation distance computes the smallest intersecting angle between two lines. The function  $f(x) = x^2/W$  is used to penalize large angle deviation but ignore small variation, where  $W$  is the weight to be determined. As illustrated in Fig. 1, perpendicular distance is the vertical distance between two lines. Parallel distance is the movement from mid-point along either line to vertically align the mid-point of two lines. If two lines are not parallel, the line with parameters  $(x_i, y_j, \theta_k)$  is first rotated to be parallel with each line before calculating perpendicular and parallel distances to that line.

### 3. Proposed Approach

The model and target images are first encoded into line sets represented by  $M = \{\bar{m}_1, \bar{m}_2, \dots, \bar{m}_p\}$  and  $T = \{\bar{t}_1, \bar{t}_2, \dots, \bar{t}_q\}$ , respectively. Each line segment is represented by its mid-point, angle and length:

$$\bar{m}_i = [x_i^m \quad y_i^m \quad \theta_i^m \quad l_i^m]^T \quad (7)$$

$$\bar{t}_j = [x_j^t \quad y_j^t \quad \theta_j^t \quad l_j^t]^T \quad (8)$$

Superscript  $m$  and  $t$  represent model line and target line, respectively. Subscript  $p$  and  $q$  represent the number of lines in model and target line set, respectively.  $x, y$  represents the mid-point coordinates

of the line segment,  $\theta$  and  $l$  represent the angle (between  $x$  axis and the line) and length (in pixels) of the line segment.  $\theta \in [0, \pi]$ .

Using (6) above, we define the forward line based Hausdorff distance between transformed model and target line set as the similarity measure between two line sets:

$$h^f(M, T) = f \text{ th} \min_{\substack{\bar{m} \in M \\ \bar{t} \in T}} d(k(\bar{m}_i), \bar{t}_j) \quad (9)$$

Where  $f \text{ th}_{x \in X} S(x)$  denotes the  $f$  th quantile value of  $S(x)$  over the set  $X$ ,  $f \in [0, 1]$ . E.g., 1/2th quantile value is the median and when  $f=1$ , Eqn. (9) is equal to  $\min_{\bar{t} \in T} d(k(\bar{m}_i), \bar{t}_j)$ .  $k(\bar{m}_i)$  is the line obtained by

applying affine transformation  $k$  to  $\bar{m}_i$ . Let  $\tau$  be the matching threshold, when  $h^f(M, T) \leq \tau$ , the transformation  $k$  brings all model lines close enough to a subset of the target line set, thus obtaining a match.  $f$  and  $\tau$  are controlled by the user to decide which transformations,  $k$ , are returned.

The matching criteria of line sets can be expressed differently in terms of fractions. It can be observed that  $h^f(M, T)$  is below  $\tau$  when at least  $f\#(M)$  of the transformed model lines lie within  $\tau$  of some target lines. We define the fraction as:

$$g(k) = \frac{\#\{\bar{m} \in M \mid \Delta(k(\bar{m})) \leq \tau\}}{\#(M)} \quad (10)$$

This is the fraction of all the model lines that, when transformed by  $k$ , lie within  $\tau$  of some image lines. The matching criterion is therefore  $g(k) \geq f$ .

Exhaustively examining all affine transformations is prohibitively expensive to perform. Even with evaluate and subdivide technique, point based approach [5] starts from a six-dimensional affine transformation space. Since a line intrinsically provides orientation, position and length information, these properties can be exploited to greatly reduce the affine search space. Since longer lines are more robust against position and orientation error [6], we utilize this property in our efficient transformation space decomposition algorithm. Let the  $MK$  represent the line set of  $K^{M\%}$  longest model lines of  $M$  and  $TK$  represent the line set of  $K^{T\%}$  longest target lines of  $T$ . We exhaustively align current model line  $\bar{m}_c \in MK$  and current target line  $\bar{t}_c \in TK$  so that, for each pair of  $\bar{m}_c$  and  $\bar{t}_c$ , their mid-point, angle and length are all equal. This is used to generate transformation parameters for translation ( $Tx_c, Ty_c$ ), rotation ( $R_c$ ) and scaling in  $x$  axis ( $Sx_c$ ) before searching for the remaining transformation parameters of shearing ( $Sh_c$ ) and scaling in  $y$  axis ( $Sy_c$ ) through

transformation space decomposition. These six transformations combined provide full affine transformations of the model line set [6].  $Tx_c$  and  $Ty_c$  is obtained by calculating the displacement of mid-point of  $\bar{m}_c$  along each axis.  $R_c$  is obtained by taking the angle difference between  $\bar{m}_c$  and  $\bar{l}_c$ . Another rotation transformation is generated by adding  $\pi$  to  $R_c$  in order to cover all rotation possibilities.  $Sx_c$  is obtained by calculating  $l'_c / l_c^m$ .

The following pseudocode outlines the object location algorithm. After aligning current model and target line, a two-dimensional transformation space with axes  $Sy$  and  $Sh$  is decomposed in increasing resolution to search for  $Sy$  and  $Sh$  parameters that satisfy the matching criterion.

#### Algorithm Line-based Cell Decomposition

Input:  $MK, TK$

For each ( $\bar{m}_c \in MK, \bar{l}_c \in TK$ ) combination

Rigid transform  $M$  so that for  $\bar{m}_c, x_c^m = y_c^m = \theta_c^m = 0$ ;

Align  $\bar{m}_c$  with  $\bar{l}_c$ , generate  $Tx_c, Ty_c, Sx_c$  and  $R_c$ ;

Initialize a list of interesting cells to cover  $(Sy, Sh)$  space;

While cell resolution  $> \tau_c$  do

Cover all interesting cells with finer resolution cells;

If any finer resolution cell contains  $Sy, Sh$  values that combined with  $Sx_c, Tx_c, Ty_c, R_c$  or  $Sx_c, Tx_c, Ty_c, R_c + \pi$  meets the matching criterion, mark the cell as interesting;

End while

Output  $Sy_c, Sh_c$  values if found;

End for

In order to evaluate whether a cell is interesting to warrant further decomposition, we compute a bound on the maximum possible value attained by  $g(k)$  for any  $k$  within a cell, thus rejecting the cell if this maximum value is below  $f$ . To compute  $\max g(k)$  we first obtain the bound on each transformed model line  $k(\bar{m}_i)$ 's  $x, y, \theta$  values so that they vary within a cube corresponding in the line based distance transform of the target line set. We then probe for the minimum line-based distance transform value within the cube.  $k(\bar{m}_i)$  can not possibly get closer than this value to some line in  $T$ .  $\max g(k)$  can be obtain by counting the number of these values that are  $\tau$  or below and divide it by the total number of lines in  $M$ , as in Eqn. (10). Since we rely on the current target line to provide structural information of objects in the target line set, we must ensure that  $\bar{l}_c$  and  $\bar{m}_c$  remain aligned for any transformation within a cell during and after the search

process. We achieve this by performing  $Sx, Sh$  and  $Sy$  before  $R, Tx$  and  $Ty$ .

## 4. Experiment and Results

The proposed affine invariant object location algorithm was evaluated on various real-world images. To avoid locating degenerative objects, the range of shearing ( $Sh$ ) is limited to  $[Sh_{\min}, Sh_{\max}]$  and parameters  $\alpha_{\min}, \alpha_{\max}$  is introduced to specify the maximum aspect ratio skew, limiting scaling in y axis ( $Sy$ ) to  $[\alpha_{\min} * Sx_c, \alpha_{\max} * Sx_c]$ . We also limit  $Sx \geq 0.3$  to avoid locating extremely small transformed models. Line based distance transform is first computed for each target line set extracted from each example image before locating objects in these examples.

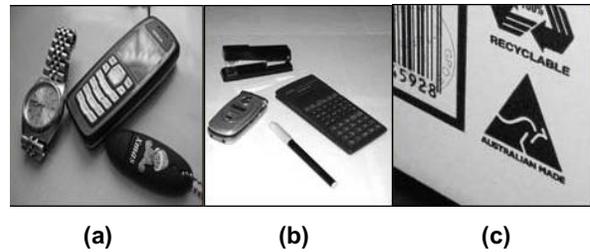


Figure 3. Test Images. (a) Cell phone example (b) Calculator example (c) Logo example

The first example (Fig. 4) locates a cell phone in an image clustered with other objects with different shapes. The parameters used were  $\tau = 2, f = 0.81, \tau_c = 0.2, Sh_{\max} = 0.2, Sh_{\min} = -0.2, \alpha_{\min} = 0.5, \alpha_{\max} = 2, K^M \% = 0.05, K^T \% = 0.7, W = 30$ . Time taken to locate the model on a Pentium 4 2.0 MHz PC was 26 seconds.

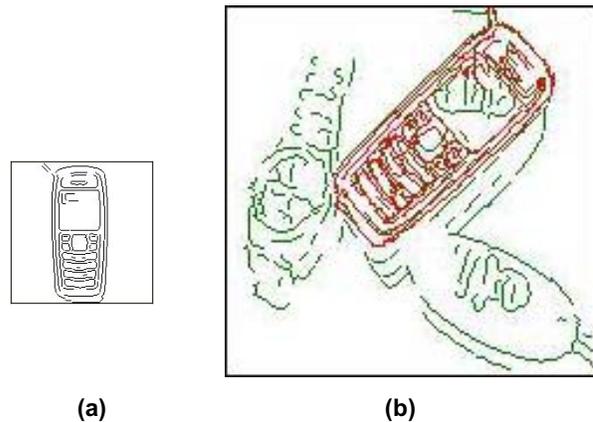
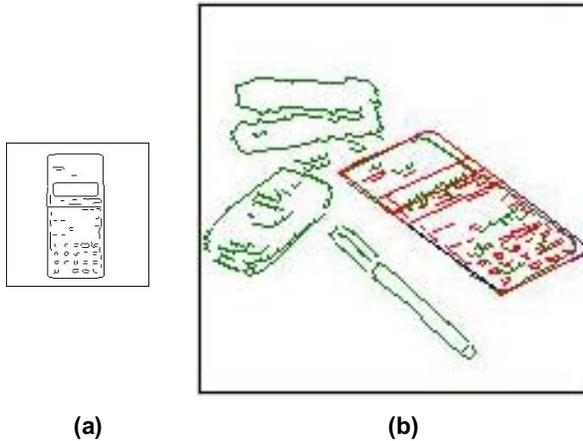


Figure 4. Locating cell phone. (a) Cell phone model (b) Located cell phone in the target line set

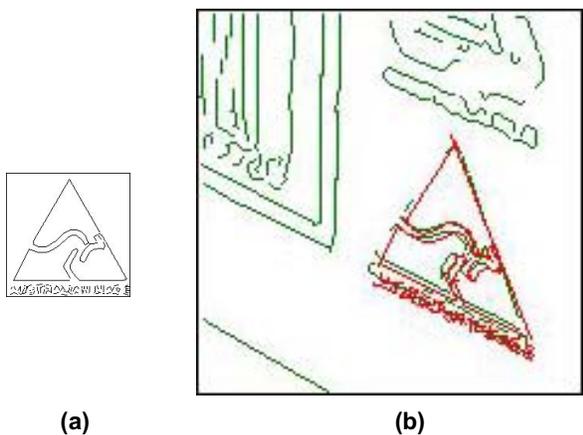
The second example shows how a calculator can be located in a scene clustered with other rectangular

objects (Fig. 5). It demonstrates that the algorithm is capable of discriminating similar shaped objects under full affine transformation. The parameters used were  $\tau = 8$ ,  $f = 0.9$ ,  $\tau_c = 0.2$ ,  $Sh_{\max} = 0.5$ ,  $Sh_{\min} = -0.5$ ,  $\alpha_{\min} = 0.83$ ,  $\alpha_{\max} = 1.5$ ,  $K^M \% = 0.05$ ,  $K^T \% = 0.7$ ,  $W=30$ . Time taken was 18 seconds.



**Figure 5. Locating calculator. (a) Calculator model (b) Located calculator in the target line set**

We also test the performance of our method on affine transformed non-rectangular objects in a clustered scene. Fig. 6 shows one such test performed on a logo. The parameters used were  $\tau = 8.7$ ,  $f = 0.81$ ,  $\tau_c = 0.2$ ,  $Sh_{\max} = 0.2$ ,  $Sh_{\min} = -0.2$ ,  $\alpha_{\min} = 0.8$ ,  $\alpha_{\max} = 2$ ,  $K^M \% = 0.05$ ,  $K^T \% = 0.8$ ,  $W=30$ . Time taken was 21 seconds.



**Figure 6. Locating logo. (a) Logo model (b) Located logo in the target line set**

Our experiments also show that the algorithm can locate only the rigid transforms of objects in the target image by setting parameters  $Sh=0$ ,  $\alpha_{\min} = \alpha_{\max} = Sx=1$ .

This results in much shorter searching time because without having  $Sh$  and  $Sy$  ranges, the need to perform cell decomposition is eliminated. Overall, the proposed algorithm is able to very efficiently discriminate between similar objects under full affine transformation in a clustered setting while reducing storage requirement by 70% on average compared to point based methods.

## 5. Conclusion and Future Work

We have proposed a new line-based methodology that efficiently locates objects under affine transformation. A new line-based distance transform is integrated to efficiently decompose transformation space in a coarse to fine manner. This methodology greatly reduces the search space of affine transformation parameters by exploiting structural information provided by line features. Furthermore, the proposed methodology demands much less storage space compared to point based approaches. Experiments with real-world model and target images clearly demonstrate the effectiveness of the proposed approach. Future work will be to develop efficient method to compute line-based distance transform and incorporate reverse verification step to reduce false matching and false rejection.

## Acknowledgements

This research was supported by the Australian Research Council (ARC) Discovery Grant DP0451091.

## 6. References

- [1] M.T. Quang and W.W. Boles, "Wavelet-Based Affine Invariant Representation: A Tool for Recognizing Planar Objects in 3D Space," *TPAMI*, 19(8): 846-857, 1997.
- [2] J. Flusser, T. Suk, "Pattern Recognition by Affine Moment Invariants," *Pattern Recognition*, 26(1): 167-174, 1993.
- [3] K. Arbter, W.E. Snyder, H. Burkhardt, and G. Hirzinger, "Application of Affine-invariant Fourier Descriptors to Recognition of 3D objects," *TPAMI*, 12(7): 752-459, 1990.
- [4] M.I. Khalil, M.M. Bayoumi, "A Dyadic Wavelet Affine Invariant Function for 2D Shape Recognition," *TPAMI*, 23(10):1152-1164, 2001.
- [5] W. Rucklidge. "Efficiently Locating Objects Using the Hausdorff Distance," *IJCV*, 24(3): 251-270, 1997.
- [6] Y. Gao and M. K. H. Leung, "Face Recognition Using Line Edge Map," *TPAMI*, 24(6): 764-779, 2002.
- [7] D. Forsyth and J. Ponce, *Computer Vision*. Prentice Hall, 2003.