Visible and Infrared Sensors Fusion by Matching Feature Points of Foreground Blobs

Pier-Luc St-Onge[1] and Guillaume-Alexandre Bilodeau[2]

LITIV, Department of Computer Engineering, École Polytechnique de Montréal,
P.O. Box 6079, Station Centre-ville Montréal (Québec), Canada, H3C 3A7,
{pier-luc.st-onge, guillaume-alexandre.bilodeau}@polymtl.ca

Abstract. Foreground blobs in a mixed stereo pair of videos (visible and infrared sensors) allow a coarse evaluation of the distances between each blob and the uncalibrated cameras. The main goals of this work are to find common feature points in each type of image and to create pairs of corresponding points in order to obtain coarse positionning of blobs in space. Feature points are found by two methods: the skeleton and the Discrete Curve Evolution (DCE) algorithm. For each method, a feature-based algorithm creates the pairs of points. Blob pairing can help to create those pairs of points. Finally, a RANSAC algorithm filters all pairs of points in order to respect the epipolar geometrical constraints. The median horizontal disparities for each pair of blobs are evaluated with two different ground truths. In most cases, the nearest blob is detected and disparities are as accurate as the background subtraction allows.

1 Introduction

By watching a calibrated rectified stereo pair[1], a human visual system is able to reconstruct the scene in 3D. But if we use a camera for the visible spectrum and another for the infrared spectrum, images from both cameras have different colors and textures, and the human visual system is just not able to reconstruct the scene. Fortunately, a computer can do it with the proper image processing.

In the recent years, infrared cameras are more affordable than ever [1]. By being more accessible, they are used in applications like video surveillance. In fact, while there could be some occlusions or false alarms in the visible spectrum, the infrared spectrum could possibly resolve these problems [2], [3]. While this is not the goal of the present research, it illustrates one of the advantages of combining visible and infrared informations. Another advantage is to exploit the two cameras to enable the evaluation of which person is farther and which person is closer. In a context of video surveillance, this could be very useful to identify and prevent crimes or strange behaviors.

In the literature, some algorithms can create dense disparity maps from two visible rectified stereo images [4]. Some other algorithms use uncalibrated infrared stereo images to create a sparse or dense disparity map [1], [5]. Unfortunatly, these methods are not useful as the corresponding points are featured

[1] There is a line by line correspondence in rectified stereo pairs.

with different values in infrared and visible images. As an example, the *Phase Congruency* algorithm [6] may find reliable feature points in any type of images, but the pairing by correlation product fails to find enough good pairs of points in infrared and visible images. So, color invariant features are needed. Shah and *al.* ([7]) have used many types of moments to describe feature points of both types of images. While this may work, their method also requires that the stereo pair have minimal disparity, so it is not very useful for the current research.

In the case of our research, both cameras are uncalibrated[2]. Nevertheless, they have to look approximately at the same vanishing point[3], and they must have about the same up vector and field of view. The cameras are distanced by at least half a meter to get significant differences between disparities.

By convention for the rest of the paper, the left and right images are the visible and infrared images respectively. All images are scaled to a resolution of 640x480 after a correction of the lens distortion. The foreground blobs are obtained by a *Temporal Averaging* algorithm like the one in [8]. Shadows are eliminated as much as possible in the left image. Finally, blobs smaller than 256 pixels of area are deleted. The remaining blobs are the input of our method.

The proposed method finds two types of color invariant feature points in each foreground blob: from an approximated skeleton (section 2) and from a simplified implementation of the Discrete Curve Evolution process (section 3). Then, feature points are further described in order to allow the pairing with a correspondence matrix comparing all possible pairs of feature points of the same type (sections 2.2 and 3.2). These are the main contributions of the paper. The outliers are filtered with two successive algorithms presented in section 4: the blob pairs determined with the pairs of points, and a specialized RANSAC algorithm. The calculation of all blobs' disparity from all remaining pairs of points is defined in section 5. Section 6 contains the experiment details and the results. Finally, the paper ends with our conclusions and suggested improvements.

2 The Skeleton

2.1 Feature Points

In a skeleton interpreted as a tree, the feature points are the vertices that are not connected by two edges (see figure 1). Here are the steps to extract the feature points in each foreground blob:

1. In the blob, delete all holes that are smaller than N pixels of area;
2. Approximate the contour of the blob with the Douglas-Peucker algorithm[4] [9]. This algorithm has only one parameter: the accuracy (in pixels);
3. Draw the approximated blob in white on a black binary image;
4. Apply a distance transform [10] on the white region of the binary image;

[2] The internal and external parameters of the stereo system are unknown.
[3] In practice, the cameras must look at the same farthest object in the scene.
[4] This algorithm, like many others, is already implemented in OpenCV.

5. With the convolution product, apply the filter $\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$ on the distance transform result;
6. Apply a threshold to get the first feature points;
7. Use the Prim's algorithm [11] to get the minimum spanning tree of the complete graph made of all first feature points;
8. In the minimum spanning tree, keep all vertices that do not have two edges.

By default, the accuracy parameter of the Douglas-Peucker algorithm is 2.5 pixels, the parameter N is about 250, and the threshold is about 4. Figure 1 gives an example of the feature points found in each image.

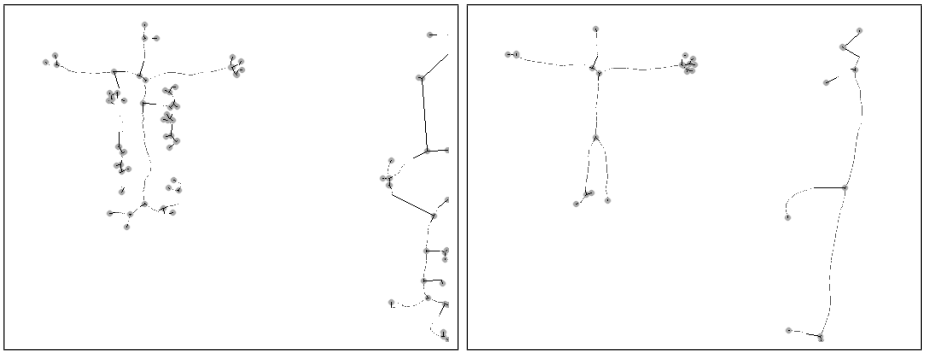


Fig. 1. Two skeletons in the left and right images. The dark grey dots are all points found after the thresholding. The big light grey points are the feature points. The black lines bind the feature points to their neighbors

2.2 Feature Points Description

All found feature points must be described in order to make pairs of similar points from the stereo images. The pairs are determined with a correspondence matrix containing the *totalScore*s of each possible pair of feature points. Of course, the corresponding points are the ones that have the highest *totalScore*s. The *totalScore* metric is the sum of four individual metrics that we have defined:

$$totalScore = scoreDist + scoreEucl + scoreEdge + scoreAngl \ . \qquad (1)$$

Here is the description of each metric:

– By normalizing the distance-transformed blobs to values from zero to one[5], each feature point is described according to its relative distance rd from itself

[5] All values in the distance-transformed image are scaled linearly to values from zero to one.

to its blob's contour. So, the first metric is

$$scoreDist = -|rd_l - rd_r| \ , \tag{2}$$

where rd_l and rd_r are the relative distances of the points in a candidate pair. Of course, l means left image, and r means right image.

– Being given that the two images have the same size, it is possible to evaluate the distance between the two points. The second metric is the sigmoid

$$scoreEucl = \frac{1}{1 + e^{-3+6d/D}} \ , \tag{3}$$

where d is the Euclidean distance in pixels between the two points, and D is the length in pixels of the diagonal of the images.

– In the minimum spanning tree, each point in the pair has one or many edges. The number of edges of the left and the right points are nb_l and nb_r respectively. So, the third metric is

$$scoreEdge = \begin{cases} 2 & \text{if } (nb_l \geq 3) = (nb_r \geq 3) \\ 0 & \text{if not} \end{cases} \ . \tag{4}$$

– For each point in a candidate pair, all point's edges are oriented from the point to its neighbor vertex in its minimum spanning tree. Then, the left point's oriented edges are compared to the right point's oriented edges in a correspondence matrix. In this matrix, the score is the cosinus of the angle between two compared edges. The largest is the cosinus, the most the two edges correspond. Being the set of corresponding edges P_e, the fourth metric for the candidate pair of feature points is

$$scoreAngl = \frac{\sum_{p \in P_e} \cos \theta_p}{\max(nb_l, nb_r)} \ , \tag{5}$$

where θ_p is the angle between the two edges in the pair p.

3 Discrete Curve Evolution

3.1 Feature Points

We use the *Discrete Curve Evolution* (DCE) algorithm described in [12] to keep the most significant points of the contour of each foreground blob. At each iteration of the DCE, instead of removing the complete set $V_{min}(P^i)$ from $Vertices(P^i)$ (see [12]), we remove only one vertex $v \in V_{min}(P^i)$. In this way, we can control how much vertices we want to keep in the final contour. We did not implement the *topology preserving DCE* process, because we wanted the code to be as simple and fast as possible. The final contours usually have no bad loops. The external contours end with 33 vertices at most (by default). The internal contours for holes larger than 256 pixels of area end with 13 vertices at most (by default). Figure 2 gives an example of the feature points found in each image.

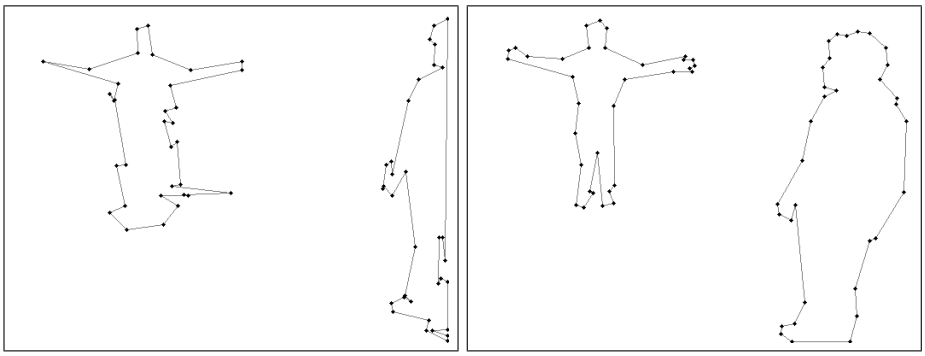


Fig. 2. Simplified contour of blobs in the left and right images. The black dots are all points found by our DCE implementation

3.2 Feature Points Description

For the DCE algorithm, the *totalScore* metric is defined differently:

$$totalScore = scoreK + scoreEucl + scoreAngl \ . \tag{6}$$

As defined in [12], the relevance measure is

$$K(\beta, l_1, l_2) = \frac{\beta l_1 l_2}{l_1 + l_2} \ , \tag{7}$$

where β is the external angle, and l_i is the length of the i^{th} edge. The metric *scoreK* is defined as:

$$scoreK = \frac{-|K_l - K_r|}{\left(\begin{cases} \sqrt{K_l^2 + K_r^2} & \text{if } \sqrt{K_l^2 + K_r^2} > 0 \\ 1 & \text{if not} \end{cases} \right)} \ , \tag{8}$$

where K_l and K_r are the relevance measures of the left and right points respectively. The metrics *scoreEucl* and *scoreAngl* are the same as in section 2.2.

4 Filtering the pairs of points

4.1 Blob Pairs Filter with Centroids

Until now, the previous methods only match points by comparing all points of the left image to the points of the right image. Of course, there are outliers that match points from two foreground blobs that do not correspond. In order to avoid these outliers, we classify each pair of points in bins representing all possible pairs of corresponding blobs; this is another correspondence matrix, and the score is the number of pairs of points. Then, it is quite easy to identify the most probable pairs of blobs. Finally, those pairs of blobs are used to compute

again the pairs of points. In fact, not only is there a correspondence matrix of $totalScore$s for each pair of blobs, but we can add another metric to both $totalScore$s in sections 2.2 and 3.2. This is done by these steps:

1. Align the left and right blobs according to their centroid;
2. Compute the new displacement $\tilde{d}$ between the left and right points;
3. Align the left and right blobs in order to get the minimal bounding box that contains them;
4. If the horizontal displacement of $\tilde{d}$ in absolute is less than a quarter of the width of the minimal bounding box, than add 1.0 to the $totalScore$.
5. If the vertical displacement of $\tilde{d}$ in absolute is less than a quarter of the height of the minimal bounding box, than add 1.0 to the $totalScore$.

By using this additional metric, we are able to get rid of some outliers that may link, for example, the head of a person to his foot (see figure 4). This is why the adaptive thresholds for $\tilde{d}$ are the quarter of the maximal width and height of both corresponding blobs. Finally, $\tilde{d}$ is *not used* in equation 3.

4.2 Epipolar Geometrical Constraint

According to [1], the epipolar geometrical constraint is represented by the equation 9, where F is the fundamental matrix, $\boldsymbol{x}_l = (x_l, y_l, 1)^T$ is a left point and $\boldsymbol{x}_r = (x_r, y_r, 1)^T$ is the corresponding right point.

$$\boldsymbol{x}_r^T F \boldsymbol{x}_l = 0 \tag{9}$$

Even after applying the blob pairs filter, there are still outliers. It is almost impossible to find a fundamental matrix F such that the equation 9 stays true for all pairs of points. Fortunately, a specialized RANSAC algorithm[6] can eliminate those outliers while accepting matric products that are not exactly zero [13]. The remaining pairs of points can be used to compute the blobs' disparity in order to evaluate the relative distance between the blobs and the cameras.

5 Blobs' disparity

For each known pair of blobs, all its pairs of feature points are grouped together. Within a group of pairs of points, one must compute the median horizontal displacement. At this stage, there are still some outliers. This is why it is preferable to use the median instead of the mean value. Because the up vector and the field of view of both cameras are about the same, it is more interesting to focus on the horizontal displacement instead of the complete vectorial displacement. Finally, the absolute value of the median horizontal displacement is the blob pair's disparity value. The figure 3 is an example of a final render.

[6] We used the implementation in OpenCV. The default maximum distance is 2.0, and the default confidence level is 0.9.

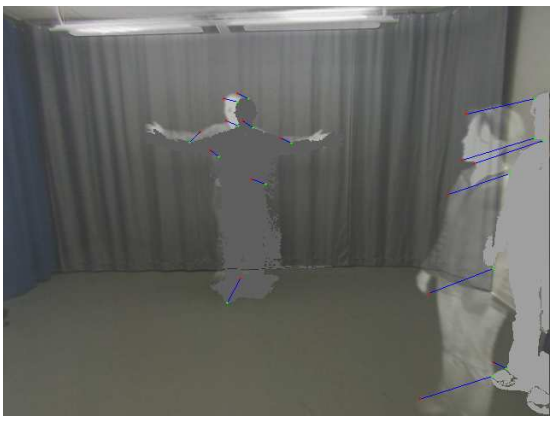


Fig. 3. A final output with all filtered pairs of points. Darker blobs are farther than pale blobs

6 Experimentation

6.1 Experiments

The first test evaluates the effect of the filters described in section 4. Then, the other tests evaluate the pairs of points with two types of ground truths:

- The first type contains identification number(s) (ID) for each significant blob at each four frames of each video. For each stereo images, the corresponding blobs have the same ID. If one significant blob is merged with another blob, the final foreground blob gets two IDs. Then, if this blob separates, the two resultant blobs get the proper unique ID. If two people are merged in one blob (identified with two IDs) in one image and if they are separated in the other image (one ID for each blob), this makes two pairs of blobs.
  With this type of ground truth, we are able to classify the pairs of blobs found with our correspondence matrix (see section 4.1): valid pairs, bad pairs (false positives) and missing pairs (false negatives).
- The second type contains informations about the distance between each blob and the left camera. The information is a relative distance without any unit. This type of ground gruth is only used to know which blob is farther than the other: it is not yet used to know how accurate are the disparities computed at section 5. The metric is then computed with the following steps:
  1. Put each blob's disparity in the vector $\boldsymbol{\alpha}$;
  2. Put each blob's ground truth relative distance in the vector $\boldsymbol{\beta}$;
  3. Do the scalar product $\gamma_1 = -\boldsymbol{\alpha} \cdot \boldsymbol{\beta}$;
  4. Sort values in $\boldsymbol{\alpha}$ and in $\boldsymbol{\beta}$, and redo the scalar product $\gamma_2 = -\boldsymbol{\alpha} \cdot \boldsymbol{\beta}$;
  5. The metric is then $\kappa = \gamma_1 - \gamma_2$.

  The negative sign for γ_i values is because the disparity values are high when the blobs are close to the camera. If κ is zero, then the order of blobs is

perfect. If κ is negative, then it only tells that our program have the wrong estimation of which blob is closer.

Finally, all tests are done by using thresholds and parameters proposed earlier in the paper. The sources are two videos of 210 frames, but we have only tested one stereo pair out of four (53 stereo pairs including the first one). The videos are recorded from a Sony DFWSX910 visible camera and a FLIR A40V infrared camera. These cameras are synchronized at 7.5 fps.

6.2 Results

The figure 4 shows the effects of applying the blob pairs filter and the epipolar geometrical constraint. It can be noted that the blob pairs filter helps to avoid coarse errors of pairing. While the RANSAC algorithm filters many outliers, it also gets rid of some inliers: we still have to find the right parameters for this algorithm. Anyway, many of the outliers are caused by the unreliable background subtraction. With a better background subtractor, we expect to get better results. For the following results, both filters are used.

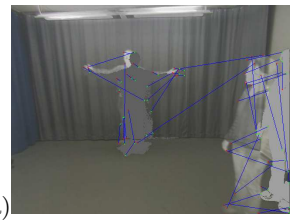
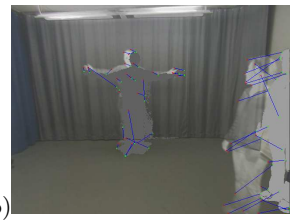
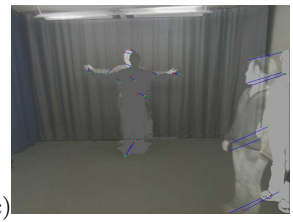
a)  b)  c) 

Fig. 4. Visual results when merging all found feature points from the skeleton and the DCE algorithms. In a), no filter is used. In b) and c), the blob pairs filter is activated. Only in c), the RANSAC algorithm is activated. In all three cases, our method properly detects that the two people are not close to each other

Table 1 shows the results with the first type of ground truth. Most of valid blob pairs have been found, so they are usable in the blob pairs filter. The bad or missing blob pairs are caused by the imperfect background subtraction (figure 5a) or by merged blobs (figure 5c). The skeleton algorithm does slightly less blob pairs errors than the DCE algorithm, probably because the DCE algorithm is more sensitive to the noise in the contour of blobs in the visible images. All these errors could be resolved by a better background subtraction or by a segmentation like in [7].

For the second type of ground truth, we have used each algorithm (skeleton and DCE) alone and then together. For each of these three cases, we got only two wrong ordering for all 53 stereo pairs. All three cases have failed on the same stereo pair when both actors are about at the same distance from both cameras. The second error of each case is in different stereo pairs, and is due to

their respective outliers (see figure 5b) and the random behavior of the RANSAC algorithm. This is why more tests have to be done to evaluate the accuracy of the disparities. Nevertheless, the examples of figure 4 show that our method is already able to say that the two people are not close to each other, and we are confident that the program is generally able to find which blob is the closest to the cameras.

Table 1. For each algorithm, the total number of found, valid, bad and missing pairs of blobs

Algorithm	Found	Valid	Bad	Missing
Skeleton	60	58	2	4
DCE	62	57	5	5

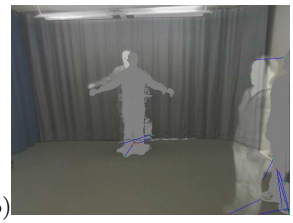
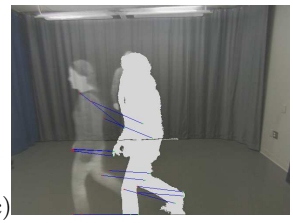
a)  b)  c) 

Fig. 5. Visual results of some anomalies. In a), the left actor's blob was cut in two during the background subtraction; the DCE algorithm and the blob pairs filter retained the smallest piece of blob. In b), too much outliers caused a wrong ordering. In c), the merged blobs caused a missing blob pair

7 Conclusion

In this work, we address the problem of stereo vision with a visible camera and an infrared camera. We have proposed two methods to find feature points from the foreground blobs: the skeleton and the Discrete Curve Evolution process. The pairs of points have been chosen with a set of metrics for each type of feature point. We have successfully filtered some outliers with the blob pairs. While the RANSAC algorithm needs to be adjusted, it already removes some outliers. Even if the proposed method is not as precise as a dense 3D reconstruction, we are able to identify which blob is the closest to the cameras.

In future works, we want to use better background subtraction algorithms. We also want to test all possible parameters to assess the performance of the proposed methods. Of course, other types of feature points will be implemented

and tested. Finally, the accuracy of blob disparities has to be tested with a third type of ground truth.

Acknowledgments

We would like to thank the Canadian Foundation for Innovation (CFI) and the Fonds québécois de la recherche sur la nature et les technologies (FQRNT) for their support with a grant and a scholarship respectively.

References

1. Hajebi, K., Zelek, J.S.: Sparse disparity map from uncalibrated infrared stereo images. In: Proceedings of the 3rd Canadian Conference on Computer and Robot Vision (CRV'06). (2006) 17–24
2. Jones, G.D., Hodgetts, M.A., Allsop, R.E., Sumpter, N., Vicencio-Silva, M.A.: A novel approach for surveillance using visual and thermal images. (2001) 9/1–919
3. Ju, H., Bhanu, B.: Detecting moving humans using color and infrared video. In: Multisensor Fusion and Integration for Intelligent Systems (MFI2003). (2003) 228–233
4. Zitnick, C.L., Kanade, T.: A cooperative algorithm for stereo matching and occlusion detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(7) (2000) 675–684 0162-8828.
5. Hajebi, K., Zelek, J.S.: Dense surface from infrared stereo. In: Workshop on Applications of Computer Vision (WACV07). (2007) 21–26
6. Kovesi, P.: Phase congruency: A low-level image invariant. Psychological Research **64**(2) (2000) 136–148 doi:10.1007/s004260000024.
7. Shah, S., Aggarwal, J.K., Eledath, J., Ghosh, J.: Multisensor integration for scene classification: an experiment in human form detection. Volume 2. (1997) 199–202 vol.2
8. Shoushtarian, B., Bez, H.E.: A practical adaptive approach for dynamic background subtraction using an invariant colour model and object tracking. Pattern Recognition Letters **26**(1) (2005) 5–26
9. Douglas, D.H., Peucker, T.K.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica: The International Journal for Geographic Information and Geovisualization **10**(2) (1973) 112–122 10.3138/FM57-6770-U75U-7727.
10. Borgefors, G.: Distance transformations in digital images. Computer Vision, Graphics, and Image Processing **34**(3) (1986) 344–371
11. Cheriton, D., Tarjan, R.E.: Finding minimum spanning trees. SIAM Journal on Computing **5**(4) (1976) 724–742
12. Latecki, L.J., Lakamper, R.: Polygon Evolution by Vertex Deletion. : Scale-Space Theories in Computer Vision: Second International Conference, Scale-Space'99, Corfu, Greece, September 1999. Proceedings. (1999)
13. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6) (1981) 381–395 358692.