View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Queensland University of Technology ePrints Archive

An Efficient Multiple Object Vision Tracking System using Bipartite Graph Matching

Matthew Rowan, Frederic Maire,
School of Software Engineering and Data Communication,
Queensland University of Technology,
PO Box 2434, Brisbane, Qld 4001,
Australia
mg.rowan@student.qut.edu.au , f.maire@qut.edu.au

Abstract

For application domains like 11 vs. 11 robot soccer league, crowd surveillance and air traffic control, vision systems need to be able to identify and maintain information in real time about multiple objects as they move through an environment using video images. In this paper, we reduce the multi-object tracking problem to a bipartite graph matching and present efficient techniques that compute the optimal matching in real time. We demonstrate the robustness of our system on a task of tracking indistinguishable objects. One of the advantages of our tracking system is that it requires a much lower frame rate than standard tracking systems to reliably keep track of multiple objects.

Key words

Multiple object tracking, Bipartite graph matching, Hungarian method, Linear programming.

1. Introduction

Visual tracking is an area of computer vision with many practical applications, and thus it is one of the field's sub-disciplines with the biggest potential impact. There are good reasons to track a wide variety of objects, including airplanes, missiles, vehicles, people, animals, and micro-organisms. While tracking single objects alone in images has received considerable attention, tracking multiple objects simultaneously is both more useful and more problematic. It is more useful because objects we want to track often exist in close proximity to other similar objects. It is more problematic because the objects of interest can touch, occlude, and interact with each other. They can enter and leave the image and we must be able to tell the objects apart. In addition, multiple object tracking must still deal with all the hard problems of single object tracking, including running at a reasonable rate and adapting to changing background conditions.

All the robust methods for single object tracking than can handle noise, occlusions, and perform some form of error correction are probabilistic. They include Kalman filtering (Foresti, 1999; Lienhart et al., 2003) and hidden Markov models (Chen et al., 2001; Couvreur, 1996).

Multiple Object Tracking requires a different approach when the objects are described using the same model. Instantiating several independent trackers for each object is not a satisfactory solution because the independent trackers tend to join together onto a single object. Methods for preventing this phenomenon include interpreting the targets as blobs which merge and split (Intille et al 1997), enforcing a minimum separation between targets (Rasmussen and Hager, 1998), the Condensation

Algorithm (MacCormick, 2000; Needham, 2001), the Probabilistic Data Association Filter (PDAF) (Bar-Shalom and Fortmann, 1988; Intille et al 1997; MacCormick, 2000) and the Joint Probabilistic Data Association Filter (JPDAF) (Schmitt, 2002; Chen et al., 2001; Rasmussen and Hager, 1998).

Surprisingly, only one paper in the literature on vision tracking uses a graph matching process for multiple object vision tracking (Chen et al., 2001). However, since the authors of this paper present results for tracking only 3 objects, we can assume that all permutations were tested to find the optimal matching.

In Section 2, we present three different matching methods. In Section 3, we compare experimentally these methods when applied to multiple object tracking.

2. A bipartite graph matching approach for multiple object tracking

One class in the bipartite graph corresponds to the expected positions of the tracked objects. The other class corresponds to the blobs from within the current image. Features such as size, position, colour, shape, and velocity can be extracted from the object blobs. The entries of the adjacency matrix of the bipartite graphs are the similarity measurements between blobs and tracked objects. Obviously, a maximum weight matching corresponds to the best assignment of blobs to objects (objects are matched to the corresponding blobs).

Unfortunately, from time to time the number of blobs found is different from the number of expected objects. To make the two classes of the bipartite graph the same size, we simply add dummy nodes to the smaller class (with similarity weight set to 0).

The matching speed is critical for a real time system. We have considered four different matching algorithms; namely, the Permutation, Hungarian, Linear Programming (LP) and Greedy Matching Methods.

The Permutation Method enumerates all possible matchings. The running time of this method is exponential in the number of tracked objects. Practically, no more than about 8 objects can be tracked this way with current hardware speeds.

The Hungarian Method finds the maximum matching in $\mathrm{O}(\mathrm{n}(\mathrm{m}+\mathrm{n}\log(\mathrm{n})))$, where n is the number of objects and $m = \binom{n}{2}$ is the number of edges in the bipartite graph (Schrijver,2003). The Hungarian method incrementally improves an initially empty matching by finding augmenting paths in the bipartite graph. Let $G = (U, W; E)$ be a bipartite graph, with colour classes U and W. Let $w : E \to R$ be a weight function on the edges. Start with an empty matching M. Given the current matching M, let D_M be the directed graph obtained from G by orienting each edge e in M from W to U, with length $\lambda_e = w_e$, and orienting each edge e not in M from U to W, with $\lambda_e = -w_e$. Let U_M and W_M be the set of vertices in U and W, respectively, missed by M. If there is an alternating path from U_M to W_M, find the shortest such path, P, and replace M with the set difference of M and the edges of P. Iterate until no path exists from U_M to W_M can be found. It can be shown that this procedure return a maximum weight matching.

The maximum matching can also be formulated as a Linear Programming problem. Let w be the weight vector indexed by edges (w_i is the weight of the i$^{\text{th}}$

edge), let x be the characteristic vector of the matching ($x_i = 1$ if the ith edge is included in the matching, otherwise $x_i = 0$), and let A be the vertex-edge incidence matrix. Then the maximum weight matching x is the solution of max $w^T x$ subject to system of inequalities $Ax \le 1$ and $x \ge 0$, where the inequalities between vectors are component wise. The remarkable result is that the solutions of linear program are 0-1 vectors that represent the maximum matchings of the associated graph.

The greedy matching algorithm starts with an empty matching and incrementally builds a good matching by greedily inserting edges one at a time. The next edge added to the current matching is the maximal weight edge that is incident to no edge of the current matching. Although the greedy matching algorithm is the fastest algorithm for building a good matching, it does guarantee to return the best matching (maximum weight matching).

3. Experimental results

The images were captured using an analogue camera input through a Matrox Meteor II frame grabber card which digitalizes the frames as a Tagged Image File Format (tiff) image using the RGB colour space. This allows for up to approximately 30 frames per second to be captured which corresponds to approximately 30 milliseconds per frame. The resolution of the frames was 320 by 240 pixels.

We will first discuss the relative speeds of the matching algorithms, then we will explore the sensitivity of the quality of the different object tracking with respect to the object matching algorithms and similarity measures. All algorithms were implemented in Matlab. The computer used was a Pentium 4 running at 1.5 GHz with 512MB of RAM. The LP optimization was done with Matlab optimization toolbox. Figure 1 shows that the permutation method breaks down rapidly. The Hungarian method performs well but was not dependable for real time use when tracking more than 10 objects, because of the variance in the running time of the Hungarian method. The LP method proves to be very fast and was able to handle perfectly the tracking of over 30 objects in real time.

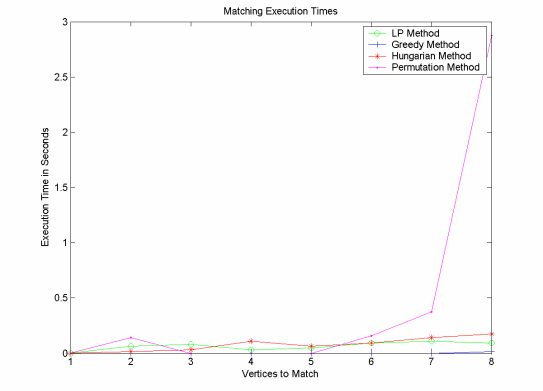

Figure 1 Comparison of the running times

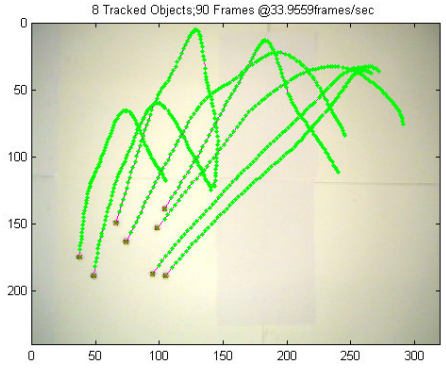

Figure 2 Trajectories over 90 frames

Although the greedy method outruns all the other methods tested, it must be stressed that the greedy method does not guarantee to return a maximum matching.

To compare the effectiveness of the matching methods, 90 frames of 8 to 20 objects moving over a non-uniform surface was captured (see example in Figure 2). The frame rate was 33.95 frames per second (29.45 milliseconds per frame). The lines displayed the objects' trajectories through the 90 frames. Each dot on the line identifies the time when a matching took place.

To test the robustness of the tracking process the matching was performed at slower frame rates. This process creates a larger distance between the objects and makes tracking much more difficult. The results of the slower frame rate testing show that the tracking can be performed in real time at slow frame rates. Overall, CPU time is saved as fewer calls to single object detection functions are made.

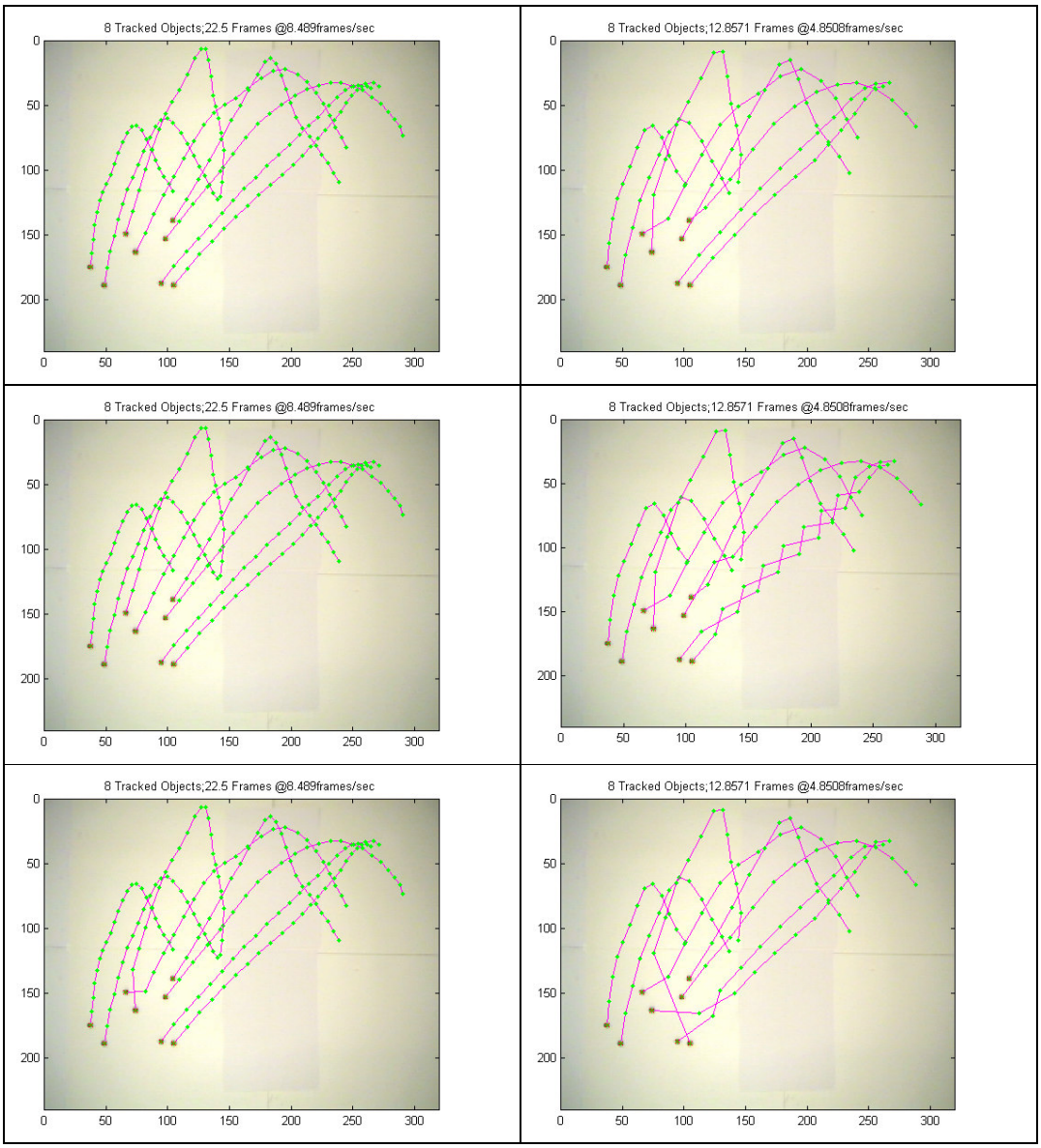

Figure 3 Comparison of matching methods. Row 1; maximum matching, predicted position similarity. Row 2; maximum matching, last position similarity. Row 3; greedy matching, predicted position similarity.

The similarity matrix was constructed using either the last positions or the predicted positions of the objects against the positions of the blobs. The objects were then matched to the blobs found in the next frame until the matching was performed over all the frames. The table of images in Figure 3 shows the results of the tracking on 8 objects. In the first row, the predicted position similarity measure was used. In the second row, the last position similarity measure was used. The tracking is better with the more sophisticated similarity measure. The third row of Figure 3 shows that the greedy matching gets confused more easily than the maximum matching.

It can be observed (see Figure 4) that the tracking of the 8 objects (marbles) is still successful at a rate of less than 4 frames per second. The first matching is incorrect for the two pairs of marbles about position (70,160) and (110,140). This is not unexpected, because for this very first matching the similarity measure used is only based on the position as no estimation of the velocity is available at that time. However, the successive matchings are correct.

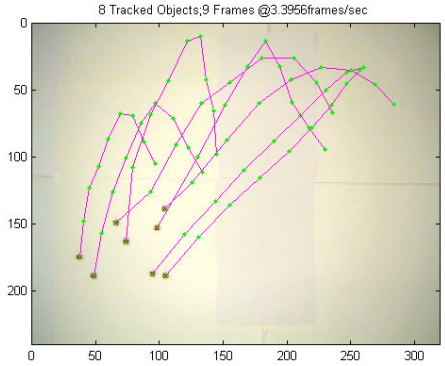

Figure 4 Low frame rate, maximum matching, predicted position similarity measure.

4. Conclusion

The experimental results presented in this paper clearly demonstrate that computing the maximum weight matching with respect to a good similarity measure yields a robust multiple object tracking system. In the experiments we conducted, the computation of the maximum matching was started from scratch at each time step. Further gain could be achieved by seeding the search algorithms with information from the previous maximum matching.

References

[1] Bar-Shalom, Y. and Fortmann, T. E. (1988) *Tracking and Data Association,* Academic Press, San Diego, CA.

[2] Chen, Y., Rui, Y. and Huang, T. (2001) 'JPDAF Based HMM for Real-Time Contour Tracking', In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1 Kauai, Hawaii, 08–14 December 2001, pp. 543-550.

[3] Chen, H.-T., Lin, H.-H. and Liu, T.-L. (2001) 'Multi-Object Tracking Using Dynamical Graph Matching', In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 2 Kauai, Hawaii, 08–14 December 2001, pp. 210-217.

[4] Couvreur, C. (1996) 'Hidden Markov Models and Their Mixtures', Faculte des Sciences - Department de Mathematiques, Universite Catholique de Louvain, Louvain.

[5] Foresti, G. L. (1999) 'Object Recognition and Tracking for Remote Video Surveillance', In *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. Volume: 9 Issue: 7, pp. 1045 -1062.

[6] Intille, S. S., Davis, J. W. and Bobick, A. F. (1997) 'Real-Time Closed-World Tracking', In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Juan, Puerto Rico, June 1997, pp. 697-703.

[7] Lienhart, R., Liang, L. and Kuranov, A. (2003) 'A Detector Tree of Boosted Classifiers for Real-Time Object Detection and Tracking', In *IEEE International Conference on Multimedia & Expo,* Microcomputer Research Labs, Intel Corporation., Baltimore, Maryland. July 6-9, 2003.

[8] MacCormick, J. and Blake, A. (2000) 'A probabilistic exlusion principle for tracking multiple objects' *International Journal of Computer Vision,* **39,** 57-71.

[9] Needham, C. J. and Boyle, R. (2001) 'Tracking Multiple Sports Players through occlusion, congestion and scale', In *Proceedings British Machine Vision Conference*, Vol. 1 School of Computing, University of Leeds, Manchester, UK, September 2001, pp. 93-102.

[10] Rasmussen, C. and Hager, G. (1998) 'Joint Probabilistic Techniques for Tracking Objects Using Multiple Part Objects', In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 1 Victoria, BC, October 13-17, 1998, pp. 191-196.

[11] Schmitt, T., Hanek, R., Buck, S. and Beetz, M. (2002) 'Probabilistic Vision-based Opponent Tracking in Robot Soccer', Institut fur Informatik, Munchen, Germany.

[12] Schrijver, A. (2003) *Combinatorial Optimization*, Springer-Verlag, Berlin Heidelberg, Germany.