

A reasoning system to track movements of totally occluded objects

Martin Antenreiter

Chair of Information Technology (CiT)
University of Leoben, Austria
mantenreit@unileoben.ac.at

Peter Auer

Chair of Information Technology (CiT)
University of Leoben, Austria
auer@unileoben.ac.at

Abstract

We present a system which is able to track objects under partial and total occlusion. The occlusion and movement reasoning uses abstract object descriptions for inference and is based on an extendable knowledge-base. The system manages various hypotheses about where objects could be and where they could reappear. It is able to deal with occlusion, noise and low-level vision failure. Our system uses a top-down approach, and is an initial step to integrate top-down and bottom-up processing of visual information.

1. Introduction

Typical vision systems integrate low-level visual cues in a hierarchical fashion, and extract relevant output from this bottom-up processing. Such processing seems inappropriate when information beyond the visual appearance needs to be taken into account, in particular when information about occluded objects needs to be maintained. To incorporate such an ability, our system fuses bottom-up visual processing with top-down reasoning and inference. The system reasons about occlusion and hiding events, and tracks occluded objects. The reasoning component maintains a hypotheses graph which is updated according to visual inputs. Using a small knowledge base for reasoning, baseline visual components are sufficient to provide the necessary input for the reasoning component.

Our system is highly modular such that several vision algorithms and an extendable knowledge-base can be easily integrated. For vision algorithms we have defined a simple interface to our system.

We tested our system on videos where a human moves rigid objects, occludes objects and hides objects within other objects. The videos are captured from a single static camera. The videos are quite noisy with a significant amount of motion blur and visual artifacts. Because of their noise level, these videos provide an interesting test suit to verify that top-down reasoning enhances the process-

ing capability of the system beyond the capabilities of pure bottom-up processing. The system's goal is to track the objects in the video — are they visible, partially occluded, or totally occluded.

1.1. Related work

There exist some occlusion reasoning systems for tracking or segmenting objects, mostly for traffic scenes or persons. Elgammal and Davis [4] use a probabilistic framework for segmenting people under occlusion. Their system operates on a pixel level, whereas our system does the occlusion reasoning on a more abstract object level. As a consequence, we do not get perfect segmentation for each frame, but a high level description of the object interactions in the video.

The approaches in [2, 8] use blobs for occlusion reasoning. A blob may consist of one or more objects, the relative depth between objects is not considered. If occlusion happens, the system merges the affected blobs into a new blob. On the other hand a blob is split, if the system is able to discriminate blobs within this blob. For an overview of tracking under occlusion in video sequences we refer to [5].

In [1] tracking results of moving objects are enhanced by reasoning about spatio-temporal constraints. The two used constraints are exclusivity and continuity. However, the whole system does only bottom-up processing.

A way to incorporate knowledge for vision systems is to use a knowledge-based approach, e.g. [7] for an aerial image understanding system, [3] for traffic monitoring and [9] for tracking for video surveillance applications. Matsuyama and Hwang [7] identified three types of knowledge in an image understanding system, and built a modular framework which integrates top-down and bottom-up reasoning uniformly. The system extracts various scene descriptions, and at the end an evaluation function selects the most promising description. The evaluation function is simple and trusts the low-level vision output more than the reasoning output. Another property of their evaluation function is that it selects the most complex scene description from the database.

In contrast, we trust the reasoning component or low-level vision components depending upon the hypothesis state; which is more flexible. While reasoning in [3, 9] is based on blobs using trajectories and velocity information, our reasoning is based on more abstract object behavior.

2. System architecture

Our system consists of three main parts: the low-level vision components, the reasoning component, and the knowledge-base used by the reasoning component. The reasoning component maintains a graph of hypotheses. Each hypothesis describes the content of a particular frame of the video, and it is linked to plausible hypotheses for the previous as well as the next frame. A hypothesis is an assumption about the states of all relevant objects. It is calculated from the vision inputs for the current frame, from one of the hypotheses for the previous frame, and from the rules in the knowledge base. Our current knowledge-base includes only some general rules about the physical world. Examples for such rules are given in Section 5.1. Communication from the low-level vision components to the reasoning component is not just bottom-up, but also includes top-down communication from the reasoning component to the vision components (e.g. to ask a vision component to look for an object at a certain location, or to update the status of a vision component). The three main parts of our system, as well as the processed video and the resulting hypotheses graph, are depicted in Figure 1. The following list describes the main steps of our system for processing a video frame:

1. Feed information from the hypotheses about the previous frame to the vision components. [Top-down processing]
2. Create new *pre-hypotheses* from the output of the vision components. These pre-hypotheses give possible object positions and the confidence of the vision components for these positions. [Bottom-up processing]
3. Apply the *construction rules* of the knowledge base to the pre-hypotheses. The construction rules construct all plausible hypotheses. [Reasoning]
4. Prune the hypotheses graph from unlikely hypotheses. This pruning is based on an evaluation function for sequences of hypotheses corresponding to the sequence of frames. [Hypotheses selection]
5. Move on to the next frame.

The next sections describe the main steps and the involved components in more detail.

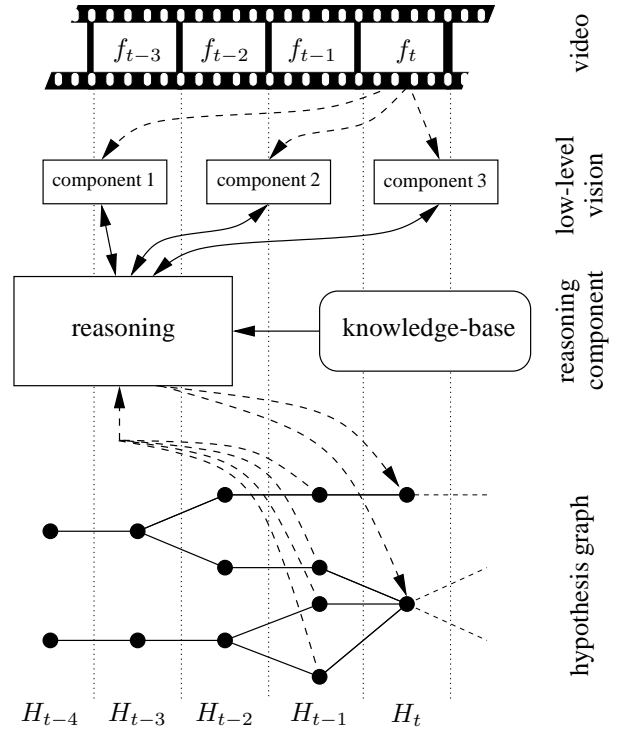


Figure 1. Reasoning system

3. Interface to the vision components

Our reasoning component “knows” more about the scene as the low-level vision components, and thus it guides the vision components. We have designed a simple interface which makes this possible. First, we need to *search* for an object in specified areas of a frame. Secondly, we need an interface for *updating* low-level vision components. In response to this we use two search instructions: the first one provides the vision component with a search window for an object, the second one queries for an object at a certain location (x, y) . The result of a search is a location with a confidence value. The confidence value indicates how sure the vision component is about the presence of the object at the returned position. A confidence of 1 indicates that the low-level vision component is very sure that object has been found. A confidence value of 0 means that the vision component failed to detect the target object. The confidence value of the low-level vision component for an object o is denoted by $detected(o)$. We use this value as a cue for either occlusion or detection failure.

Additionally, the search in a window can return a list of possible object locations. Then the reasoning component is able to choose a sub-optimal object position — from the point of view of the vision component — if this position is better supported by the overall hypothesis.

After constructing a hypothesis, the reasoning compo-

ment may use the *update* instruction to modify the status of a low-level vision component, in particular telling the low-level vision component about the positions of objects as inferred by the reasoning component. This update is crucial to insure that the status of the vision components comply with the positions of objects in the constructed hypotheses.

3.1. Vision components used in our system

The current system uses an adaptive template tracker for each target object. The tracker uses two templates, one from the initialization frame and the other one as a running average over the most recent frames. These two templates are weighted and form the template for the matching. As a confidence value we use normalized cross-correlation [6].

For hand detection we use a simple skin detector. After the detection morphological operations are applied and the areas of the blobs are calculated. Simple rules remove noise and increase stability. We treat every blob of a certain size as a hand with confidence 1.

4. Hypothesis representation

A hypothesis consists of a set of object descriptions. Visible objects are stored directly in such a hypothesis. Occluded objects are stored within the object description of the occluding object.

4.1. Object representation

The high level object description uses a simple bounding box for the object.¹ Every object stores the confidence value *detected(o)* from the low-level vision component. Additionally, every object has two state variables: one describes the move state, the other one describes the visibility state. Depending on these states, the high-level reasoning framework controls the low-level vision component for the object.

Finally, each object maintains a list of other objects which are behind or inside the object.

5. Reasoning

The reasoning component has to maintain the hypotheses graph, which is a layered graph with one layer for each frame of the video. The results from the low-level vision components produce pre-hypotheses with possible object locations. The rules from the knowledge-base update the states in the object descriptions and possibly split hypotheses into several plausible hypotheses if the correct states

¹Currently we are investigating possibilities to use the convex hull. Such a description would better fit the object boundary.

cannot be inferred with sufficient confidence. The processing of a hypothesis is finished if no further rules can be applied.

5.1. Construction rules

In this section we describe the rules for hypothesis construction. The main idea is to split a hypothesis into two or more plausible hypotheses, if the correct hypothesis cannot be inferred. What can be inferred mainly depends on what can be observed. For example, consider our test suite with only one static camera. Thus the distance of objects from the camera cannot be observed. If two bounding boxes intersect, it cannot be decided ad hoc which object is in front of the other object. Thus the current hypothesis is split in two and each possibility is represented by one of the hypothesis.

1. If the bounding boxes of two objects intersect, then split the hypothesis into two. One hypothesis states that the first object is in front, the other hypothesis states that the second object is in front.
2. If an object is occluded and a hand intersects its bounding box, then generate an additional hypothesis where the object is attached to the hand. In the following frames search for reappearance of the object along the trajectory of the hand.
3. If an object is totally occluded by one object, then we generate two hypotheses. One hypothesis states that the object does not move and reappears at the old position. The other hypothesis states that the occluded object is inside and reappears along the trajectory of the bigger object.
4. If an object is occluded, and the object gets more and more occluded, then we should not trust the vision component in every case. We split the hypothesis in two. One hypothesis assumes that the object does not move, the other one trusts the output of the vision component.
5. If the vision component returns two or more possible object positions with similar confidence for an object, then generate one hypothesis for each plausible object position.
6. An object is set to the state 'visible', if no occlusion is detected or if the object is found along the trajectory of an occluding object or hand.

6. Evaluating hypotheses

After processing the construction rules, the likelihood of each hypothesis is evaluated. The evaluation

function $Q(\mathcal{H}|\mathcal{F})$ evaluates a hypothesis path $\mathcal{H} = \langle h_1, \dots, h_m \rangle$ from the root hypothesis at frame 1 to a hypothesis for the current frame where $\mathcal{F} = \langle f_1, \dots, f_m \rangle$ is the corresponding sequence of frames. It is a quality measure of a hypothesis path compared to other paths. A simple method for calculating the quality measure Q is to sum the quality measures q for each hypotheses on the path,

$$Q(\mathcal{H}|\mathcal{F}) = \frac{1}{m} \sum_{i=1}^m q(h_i|f_i). \quad (1)$$

Before we define q , we define an inconsistency function ι and an object confidence function c .

The inconsistency function ι combines the confidence value $detected(o)$ of the low-level vision component with the relative hide value $relhide(o)$, and should detect inconsistency between these values.

$$\iota(o|h_t, f_t) = \left| -\frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2} detected(o|h_t, f_t) + \frac{\sqrt{2}}{2} relhide(o|h_t) \right| \quad (2)$$

We assume, that if the low-level vision confidence is small then the occluded object area should be large. This assumption is a rough approximation, but it is good enough for combining the two object values in practice. The function $relhide$ is a measure of what percentage of the bounding box from object o is occluded, ranging from 0 to 1. The inconsistency ι is small if the low-level vision confidence is small and the occluded object area is large, or vice-versa. This definition of inconsistency is a rough approximation, but works well on our test suite. The function ι returns the normal distance of a query point from the straight line defined by the two points (1, 0) and (0, 1); therefore the values are between 0 and $\frac{\sqrt{2}}{2}$. Figure 2 shows the inconsistency values of the function ι . However, we note that the inconsistency of our template tracker for an object does not only depend on the trackers output characteristic, but also depends on the occluding object: if the occluding object looks similar to the occluded object, then the low-level vision confidence might not decrease for the occluded object.

The plausibility function p maps the inconsistency value to a value between one and zero. For objects which are not totally occluded we define

$$p(o|h_t, f_t) := -\frac{2}{\sqrt{2}} \iota(o|h_t, f_t) + 1. \quad (3)$$

A different definition of plausibility is needed if an object is totally occluded. It would be misleading if a plausibility value of 1 were assigned to totally occluded objects, since a visible object will not typically receive a plausibility value

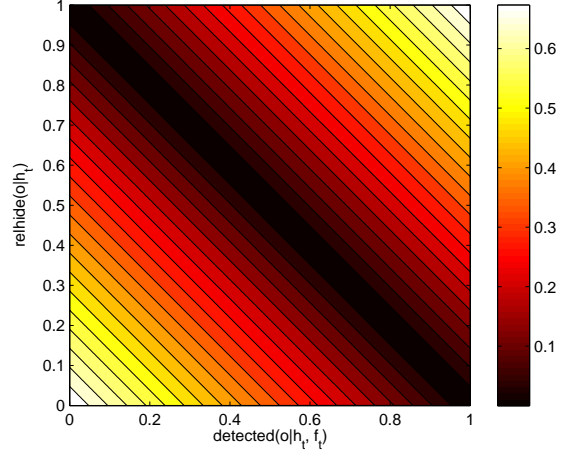


Figure 2. Inconsistency function

of 1 due to noise, illumination changes, etc. Then a hypothesis with a total occluded object would have higher plausibility than a hypothesis where the same object is partially visible. Therefore we bound the plausibility of a total occluded object by its maximal plausibility when it is visible:

$$p(o|h_t, f_t) = \max_{\substack{h \in H_t \\ o \text{ is visible}|h}} p(o|h, f_t)^1 \quad (4)$$

where $H_t = \{h : h \text{ is a hypothesis for frame } f_t\}$ is the set of all hypotheses for frame f_t .

The quality measure for a hypothesis of frame f_t is then the sum of all normalized plausibility values,

$$q(h_i|f_t) = \frac{1}{\#\{o : o \in h_i\}} \sum_{o \in h_i} \frac{p(o|h_i, f_t)}{\max_{h \in H_t} p(o|h, f_t)} \quad (5)$$

6.1. Pruning rules

Pruning strategies are implemented as rules and use the results of the evaluation function. We reject a hypothesis if an object matches one of the following implemented pruning rules:

1. Reject a hypothesis if an object has state 'visible', does not move, and has a low confidence value from the low-level vision component. (We assume, that low-level vision works well for static objects.)
2. Reject a hypothesis if an object has state 'visible' and has very low confidence. (A visible object must have at least a minimum confidence value, to be valid.)

¹If there does not exist a hypothesis $h \in H_t$ such that object o has state 'is visible', then we set $p(o|h_t, f_t) = 1.0$.

- Reject a hypothesis if an object's state was 'non-visible', the occluded area is very small, and the confidence is low. (If an occluded object reappears, the low-level vision should recognize the object and provide an appropriate confidence value.)

The thresholds for these rules were selected in a very conservative manner, such that it is very unlikely — across a wide range of videos — that a pruning rule deletes a correct hypothesis. Pruning is used only for efficiency to reduce the number of nodes in the hypotheses graph.

7. Results on some videos

Figure 3 shows a frame from our test sequences. A human moves three cups on a desktop. The cups are occluded by the human using his hands or by other cups. Sometimes one cup is total hidden as a result of a cup which is in front and due to one or more hands. A cup can also be inside another cup and can therefore reappear elsewhere. The green/white bounding boxes show the location of the

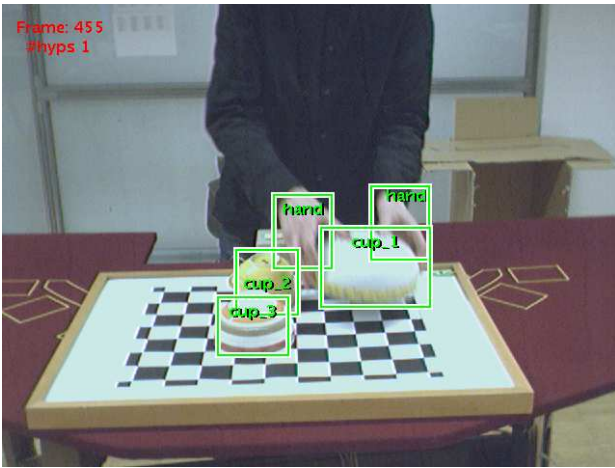


Figure 3. A test sequence with three cups. The best hypothesis assumes that 'cup 2' is behind 'cup 3'.

objects according to the most likely hypothesis. The object label is written inside the corresponding bounding box. The line position of the label indicates the relative depth between the objects. If the label is written near the upper bounding box line, then the object is in front of all the other intersecting boxes. The label is bracketed if the occlusion reasoning does not use the low-level vision input. Question marks indicate that the reasoning has assumed a low-level vision failure. The best low-level vision input is drawn with a dashed box, if the reasoning supposes an error.

Figure 4 shows four frames from the video. The hypothesis for frame 486 shows that the reasoning component does

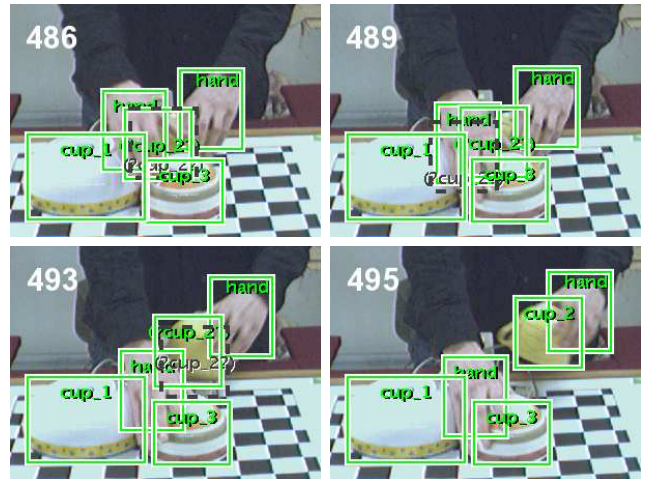


Figure 4. Sequence with occluded cup and low-level vision failure.

not trust the low-level vision component for cup 2, because of the high occlusion. The relative depth between the cups is correct. The next frame shows that the tracker is confused by the hand. The best detection result is on the right hand. The hypothesis assumes that the yellow cup is attached to the left hand, therefore the tracker search window is set along the trajectory of the left hand. In frame 493 the tracker and hypothesis position of cup 2 converge to the same position. Two frames later, the reasoning system believes that the tracker of cup 2 is once again trustworthy, and therefore the system uses the position of the vision component.

The video sequence in Figure 5 shows a total occlusion and re-detection event. The big cup 1 hides cup 3. After that the two cups are moved together; during that situation the system shows the estimated position of cup 2. The human wants to trick the system and moves the yellow cup in front of cup 1. The hypothesis assumes the following relative depth order: cup 2, cup 1, cup 3. From frame 605 to 620 cup 3 is released, but due to the high occlusion there exists very little visual evidence for the fact, that cup 3 is behind the yellow cup. Therefore the best hypothesis assumes that cup 3 is inside cup 1. We have not defined special rules for how cups can release other cups. After the actor moves cup 2 away from cup 3, the low-level vision component provides good confidence values at a previous position of cup 1. The best hypothesis with correct depth values is shown in the last picture.

8. Discussion and Conclusion

Although we defined a quite simple interface to the low-level vision components, it was difficult to find vision com-

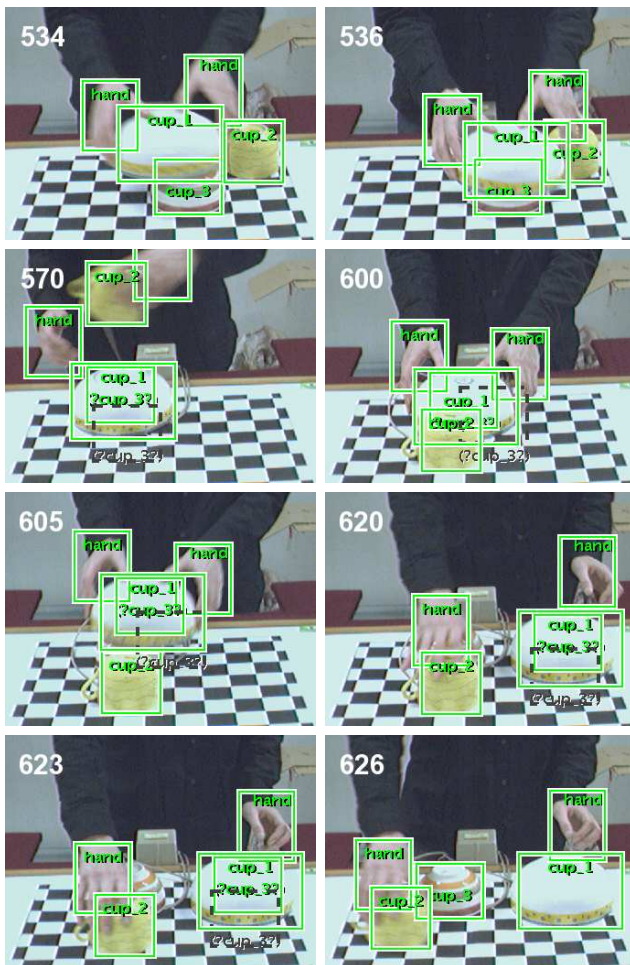


Figure 5. Sequence with hide event

ponents which could be made compatible with this interface. Mostly vision algorithms give an output as the only possible truth, and only few vision algorithms return a quality measure. Without quality measure or confidence values it seems difficult to build a cognitive vision systems.

Currently, our system is not capable of real-time applications. The average processing time per frame is 1.7s on a Pentium 4 machine with 2.4GHz. Most of the processing time is spent within the low-level vision components. The hand detector with its morphological operations is the slowest component. To overcome the problem of too many low-level vision queries, we want to incorporate a tolerance for object positions. It seems futile to generate many low-level queries for the same object with similar descriptions at similar positions. The main drawback of our reasoning component is its need for a predefined knowledge-base. We view this as an investigation about what knowledge is needed for vision system with reasoning. The next major step is to add an adaptive component which can modify and learn the

knowledge-base from examples. Another subject for improvement is the evaluation function of our system. We will modify the evaluation function such it takes the smoothness of object trajectories into account.

In this paper we described our top-down approach for occlusion reasoning with a predefined knowledge-base. The system can deal with total and multiple occlusions, and is robust against low-level vision failure. The modular architecture makes modifications of the system easy. The system was tested successfully with videos where rigid objects were both partially and totally occluded.

9. Acknowledgments

The first author acknowledge the discussions with Thomas Jaksch, Bernhard Jung (FWF-JRP S9106-N04), and Wolfgang Ponweiser (FWF-JRP S9106-N04). I am grateful, that Thomas Jaksch translated the Matlab code of vision components to a faster C++ code. The videos were provided from our partner CURE (FWF-JRP S9107-N04).

This work was supported by the FSP/JRP Cognitive Vision of the Austrian Science Funds (FWF-JRP S9104-N04 SP4) and by the IST program of the European Community, under the PASCAL Network of Excellence, IST-2002-506778.

References

- [1] B. Bennett, D. R. Magee, A. G. Cohn, and D. C. Hogg. Using spatio-temporal continuity constraints to enhance visual tracking of moving objects. In *ECAI-04*, pages 922–926, 2004.
- [2] F. Brémond and M. Thonnat. Tracking multiple non-rigid objects in video sequences. *IEEE Transaction on Circuits and Systems for Video Technology Journal*, 8(5), Sept. 1998.
- [3] R. Cucchiara, M. Piccardi, and P. Mello. Image analysis and rule-based reasoning for a traffic monitoring system. *IEEE Transactions on Intelligent Transportation Systems*, 1(2):119–130, June 2000.
- [4] A. M. Elgammal and L. S. Davis. Probabilistic framework for segmenting people under occlusion. In *ICCV*, pages 145–152, 2001.
- [5] P. Gabriel, J. Verly, J. Piater, and A. Genon. The state of the art in multiple object tracking under occlusion in video sequences. *Advanced Concepts for Intelligent Vision Systems*, pages 166–173, 2003.
- [6] J. P. Lewis. Fast template matching. *Vision Interface*, pages 120–123, 1995.
- [7] T. Matsuyama and V. S. Hwang. *SIGMA: A Knowledge-Based Aerial Image Understanding System*. Perseus Publishing, 1990.
- [8] S. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking groups of people. *Computer Vision and Image Understanding*, 80(1):42–56, Oct. 2000.
- [9] L. D. Stefano, M. Mola, G. Neri, and E. Varani. A rule-based tracking system for video surveillance applications. 2002.