Frame-rate Multi-body Tracking for Surveillance*

Terry Boult tboult@eecs.lehigh.edu
R. Micheals, A. Erkan, P. Lewis, C.Powers, C. Qian, and W. Yin
Vision and Software Technology Lab, EECS Department Lehigh University

Abstract

Video surveillance is watching an area for significant events. Perimeter security generally requires watching areas that afford trespassers reasonable cover and concealment. Almost by definition such "interesting" areas have limited visibility distance. These situations call for a wide field of view, and are a natural application for omni-directional VSAM.

This paper summarizes our ongoing efforts on developing an omni-directional tracking system. We begin with a few examples and then discuss the background and application constraints. We end with a summary of our approach and its novel components.

1 Examples & Background

The paracamera system captures omni-directional video that allows one to generate geometrically correct perspective images in any viewing direction. Figure 1 shows an example.

While unwarping in multiple directions and then doing tracking on the perspective images would be possible, it would add considerable expense. Therefore, we are working directly in the complex geometry of the paraimage.

While it is acceptable to run tracking algorithms directly on the paraimage, it is not the best way to show the targets to human users. The system provides the user a collection of windows that contain perspectively corrected images. While any number of windows are allowed, we generally use between 1 and 6 depending on the anticipated number of moving objects. The viewing direction within these windows can be controlled via the mouse, or set automatically such that the perspective windows track the N most "significant" targets.

Note the "spatial resolution" of the paraimage is not uniform. While it may seem counter intuitive, the spatial resolution of the omni-directional images is *greatest* along the horizon, just where objects are most distant. While the process scales to any size imager, the current systems use NTSC (640x480) or PAL (756x568) cameras. If we image the whole hemisphere, the the spatial resolution along the horizon is $\frac{240\text{pixels}*2*\pi}{360\text{degrees}} = 4.2\frac{\text{pixels}}{\text{degrees}}$ (5.1 PAL) which is 14.3 arcminutes per pixel (11.8 PAL). If we zoom in on the mirror, cutting off a small part of it, to increase the imaged mirror diameter to 640 pixels (756 PAL), we can achieve 10.7 (6.6 PAL) arcminutes per pixel.

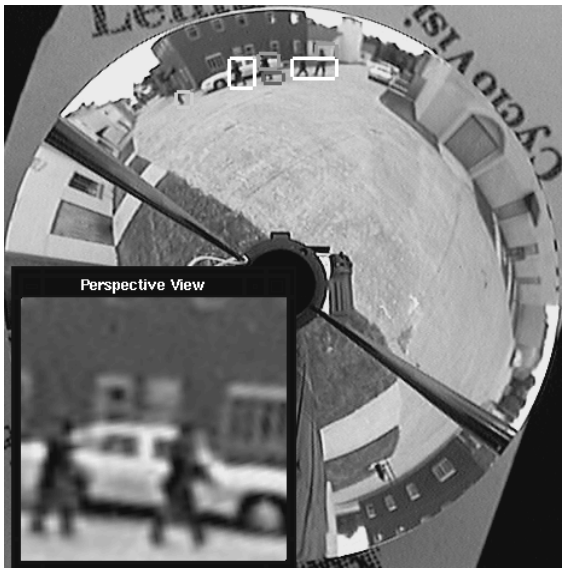

Figure 1: Tracking system with a single perspective "target" window.

As a point of comparison, let us consider a traditional "wide-angle" perspective camera. It would take 3 cameras with a 150° horizontal FOV to watch the horizon, but of these each would have $\frac{640}{150\text{degrees}} = 4.2\frac{\text{pixels}}{\text{degrees}}$, i.e. about the same as the paracamera. Clearly, the traditional cameras would need more hardware and computation.

*This work supported in part by DARPA VSAM program.

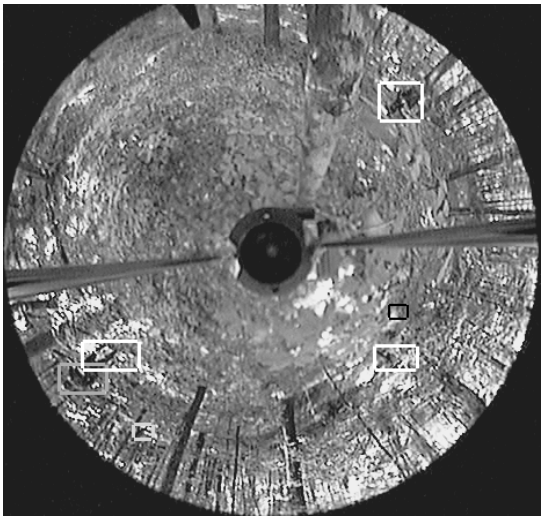

Figure 2: Tracking soldiers moving in the woods at Ft. Benning GA. While the lack of motion information and loss of resolution in printing has obscured the details, each box is on a moving target.

Every surveillance system must consider the trade-off between resolution and field-of-view. The paracamera's unique design yields what may be a new pareto optimal design choice in the resolution/field-of-view trade-off. We have the horizontal resolution of a 150° camera but cover the full 360° of the horizon.

With a wide field of view, objects to be tracked will cover only a small number of pixels. With 4.2 pixels per degree, a target of dimension 0.5m by 2.0m, at 50m will be approximately 2 pixels by 8 pixels, i.e. 16 pixels per person. At 30m, it yields approximately 32 pixels per person, presuming ideal imaging. Realistic tracking in a such a wide field of view requires the processing of the full resolution image with a sensitive yet robust algorithm.

Tracking systems abound, e.g., see [Flinchbaugh and Olson-1996, Intille *et al.*-1997, Wren *et al.*-1997] and our system draws ideas from these and many other papers. Outdoor operation in moderate to high cover areas restricts the techniques that can applied. Furthermore, we are looking for soldiers not tracking pedestrians in a store or parking lot. Some constraints, and their implications for our systems include:

- Correlation, template matching and related techniques cannot be effectively used because in a paraimage, image translation is a very poor model; objects translating in the world undergo rotation and non-linear scaling.
- The lighting is unconstrained. We must handle sunlight filtered through trees and intermittent cloud cover. (We are not considering IR cameras, yet).
- Targets will probably use camouflage to blend in, so color is not likely to add much information. Figure 2 shows an example scene with solders in the woods.
- Trees/brush/clouds all move. The system must have algorithms to help distinguish these "insignificant" motions from target motions.
- Many targets will move slowly (less than $\frac{1}{60}$ pixel per frame); some will move very slowly. Some will try very hard to blend into the motion of the trees/brush. Therefore frame-to-frame differencing is of limited value.
- Targets will not, in general, be "upright" or isolated. Thus we have *not* added "labeling" of targets based on simple shape/scale/orientation models.
- Targets need to be detected quickly, when they are still very small and distant.
- Since field use will require ruggedized low-power units, we should use generic computing hardware.

2 LOTS: Lehigh Omnidirectional Tracking System

For the past year we have been working on developing a system that can work within the constraints discussed in the previous section. Note that most of these are generic problem constraints and are not dependent on the geometry of the paraimage. Thus, the algorithms could be applied (with some minor changes) to regular perspective images. We will briefly cover some of the uniqueness of the algorithms and the techniques that allow full resolution processing at full 30fps frame rates on standard PC hardware.

Like many systems, our processing starts with change-detection based on subtraction of a "background" image. Because a stationary omnidirectional cameras does not need to pan and tilt to cover a viewing large area, it has opportunities for developing strong background models. Our "background subtraction" has three distinctive features: its adaption speed, its background modeling, and its thresholding method.

Most background based systems use temporal integration to adapt to changing lighting. Many also benifit from it "streaking" effect which, for large fast moving targets, increases connectivity and apparent size. However, because of the very gradual

image change inherent with our targets slow speed and small size, we use a very slow temporal integration. The system supports pixel updates with the *effective* integration from 25% of the new image (very fast integration) down to 0.006% of the new image. For example, given a target that differs from the background by 32, and a "threshold" of 16, this takes between 2 to 4000 frames (1/15 of a second to 2+ minutes) for the target to become part of the background. For gradual lighting changes, even the slowest of these is sufficiently fast; for rapid lighting changes, e.g., the sun going behind a cloud, alternative heuristics are applied. For the sake of both speed and numerical accuracy the system does not update the background images every frame, rather it reduces the rate at which the background is updated, e.g. an effective integration factor of .006% is achieved by adding in 1/32 of the new frame, every 512 frames. To further prevent targets from blending into the background, the pixels within a detected targets are updated one forth as often. Side-effects of this approach are that some false alarms tend to persist, and when objects that are stationary for a long time depart, they leave behind long-lasting ghosts.

The second significant feature of our background technique is that there is not a single background model, but 2 different backgrounds models, i.e. pixel can have 2 different "backgrounds". This is a significant advantage for ignoring real but contextually insignificant motions such as moving trees/brush. When the trees move they occlude/disocclude the scene behind them and the system ends up building models for both backgrounds. Currently we acquire the second background model by an initial batch learning with interactive supervised learning when false-alarms occur. We are beginning to look into more automatic methods. If false alarms occur during processing, the user may request that particular regions update their secondary background model to prevent further false alarms. The testing against the secondary background adds very minimal cost because it is only consulted when the object does not match the first background. The disadvantages are the additional memory requirement and the complexity in the learning algorithms.

In addition to having two backgrounds, the system has two thresholds. The first, a global threshold, handles camera gain noise and can be dynamically adjusted. The second, a per-pixel level threshold, handles the inherent variability of the scene intensity at a point. The threshold used in change detection is the the sum of these two components. The actual thresholding has both an MMX optimized and non-MMX implementation.

To keep the subsequent processing fast, the thresholding process keeps pointers to the initial and final pixels, per row, that are above threshold. Rows with nothing above threshold (usually 80% or more of the image) are skipped in subsequent processing. Because we expect there to be only a small collection of pixels above threshold, the thresholding phase checks this assumption. If it is violated, it is probably a rapid lighting change and the system tries a few heuristics to compensate.

After thresholding, the system needs to find connected components. Keeping this process fast is aided by two techniques. First, only pixels between each row's initial and final above threshold pixel are processed. The connectivity code also has special cases for when the entire previous row was empty. The second, and more significant speedup, comes from a reduction in resolution. The thresholding process also builds a lower resolution image of those pixels above threshold. The pixels in the parent (low resolution) image maintain a count of how many of the children were above threshold. Since resolution is reduced by a factor of 4 in each direction, the parent image contains values between 0 and 16.

The connected component phase is only applied to the parent image. In addition to the speedup, this also has the effect of filling in many small gaps. The gap filling is spatially varying; the maximum distance "neighbors" varies between 4 and 8 pixels.

After the connected components processing, the detected regions are subjected to area thresholding to remove noise regions. The area thresholds which are applied per region use the accumulated pixel counts from the parent image. This allows the system to detect (and retain) a human targets at 50m, i.e., a 2 pixels by 8 pixels region in the full resolution paraimage.

After the connected components, we have a collection of regions that are different from the background. The tracking phase attempts to connect these regions to those from previous frames. The simplest, and most common, aspect of this association occurs when the current regions is "on top of" the previous region. The system actually solves this part of the association while it is doing its connected components labeling. The labeling looks at both the current parent image as well as the parent image from the past frame. Objects that are connected in space-time are labeled with the same label they had in the past frame. (Segmentation of individuals within a closely

packed group is not, currently, being investigated.)

After handling the spatio-temporal connected regions, only a small number of regions remain. Therefore, the system can spend considerably more time trying to match up these regions. It looks to merge new regions with near-by regions that have strong temporal associations. It also looks to connect new regions with regions that were not in the previous frame but that had been tracked in earlier frames and disappeared. Both of these more complex matchings use a mixture of spatial proximity and feature similarities and are a major issue in our ongoing efforts.

For each tracked obect, the system computes and displays via color encoding a heuristic confidence measure that is based on many contributing factors including the objects size, contrast, how long it has been tracked, and how fast/far it has moved. This provides an easy way for users to crudely adjust their probability of detection versus false-alarm-rate by demanding only higher confidence targets.

As part of our VSAM project, and in an effort to begin evaluation of omnidirectional imaging for SUO-SAS, we made 3 trips to Ft. Benning to collect omni-directional image data. This data will be used throughout the 1998-1999 time frame to develop, tune, and evaluate our omni-directional tracking algorithms. Approximately 70 hours of omnidirectional video was collected. Data includes both significant amounts of "targets" and empty scenes for false-alarm evaluation. Atmospheric conditions include light rain, partly sunny and windy to sunny with light breeze. Limited copies of data are available upon request from tboult@eecs.lehigh.edu.

Researchers at the Institute for Defense Analysis have done some preliminary analysis of the tracker, as of Aug 1998, over different scenarios. The results were approximately 95% detection percentages (range from 100% down to 87%) and a false-alarm-rates ranging from .15FA per min to 1.7FA per min. The scenarios evaluated included a short indoor segments, two urban/street scenes, two different wooded settings, a town edge (half dirt/sandy and half urban) and a sniper in a grass field. (These evaluations did not include the the use of any confidence measures, nor did it allow for incremental learning or adaptive feedback on false alarms.)

Part of their feedback was that our current false alarm rate is too high. A large fraction of our current false alarms are small to moderate sized location with lighting related changes, e.g. small sun patches filtering through the trees or shadows. In a wide field of view, many of these appear very much like a person emerging from occlusion. We are currently investigating techniques, in addition to the current adaptive (supervised) learning, to label these as insignificant events or at least to reduce their "confidence" without impacting probability of detection for real targets. We are also addressing a number of minor user interface issues.

A final component of our ongoing efforts is the multi-camera coordination and a fully networked system. With this extension, the targets are tracked in local sensor processing units (computer/camera pairs) coordinated by an overall control unit (OCU) which tracks results, handles target hand-off and does integration of information in 3D. Target information and significant video clips are displayed by a networked display controller (NDC). The goal is to have one networked computer connected to 5-20 paracameras with all of the "events" being viewed on NDC. One of the design constraints in our development was the ability of the protocol to scale to large numbers of sensors each with a large number of targets while not saturating the network. The design underwent a number of iterations and in the spring we coordinated with CMU on the design of the current VSAM protocol which incorporated key ideas from both the original Lehigh and CMU designs.

Our tacker is running under Linux using MMX enabled processors. The code described herein runs at 30fps on a 233 Mhz K6 with 32MB of memory and a PCI frame-grabber. We have demonstrated a smaller system based on a 166MMX in a Compact-PCI housing (12x5x5) that tracks at 15fps. (We re now upgrading that to a 233MMX in a rugged enclosure.) We are also porting the tracker to our augmented Remote Reality "wearable" (a low-power 133MMX based system), see [Boult-1998].

References

[Boult, 1998] T. Boult. Remote reality via omnidirectional imaging. In *Proc. of the DARPA IUW*, 1998. These proceedings.

[Flinchbaugh and Olson, 1996] B. Flinchbaugh and [0] T. Olson. Autonomous video surveillance. In *25th AIPR Workshop: Emerging Applications of Computer Vision*, May 1996.

[Intille *et al.*, 1997] S. Intille, J. Davis and A. Bobick. Real-time closed-world tracking. In *IEEE CVPR*, pages 697–703, 1997.

[Wren *et al.*, 1997] C.R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Tran. on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997.