Development of Visual Tracking Algorithms for an Autonomous Helicopter

David E. Cardoze
Ronald C. Arkin

Mobile Robot Laboratory, College of Computing
Georgia Institute of Technology, Atlanta, Georgia, 30332
cardoze@cc.gatech.edu, arkin@cc.gatech.edu

Abstract

A visual target designation and tracking system is being developed within the context of the Autonomous Scout Rotorcraft Testbed Project at Georgia Tech. This paper describes both the algorithms and the hardware being used for this purpose by the Mission Equipment Package Technology Area Team. Preliminary results using two simple tracking algorithms are presented.

1. Introduction

This paper presents the approach being used to achieve target designation and tracking capabilities for a walking person in an outdoor environment from an autonomous helicopter platform. The overall project, referred to the Autonomous Scout Rotorcraft Testbed (ASRT), has as a goal the construction of a vertical takeoff and landing (VTOL) air vehicle with the ability to conduct reconnaissance and surveillance missions. A typical mission will require the ASRT vehicle to take off and fly to an operator designated search area. Once in the search area, the vehicle will follow commands from an operator located at a remote ground station to search for a human-sized moving target. Upon designation of the target the helicopter will then switch to an autonomous tracking mode while providing the person's position and stabilized imagery to the ground station. Finally, the vehicle must fly to a prescribed location and perform a vertical landing when commanded to do so.

Our approach to target tracking and designation involves the use of various visual sensors for differing conditions, including a low-light level camera for night operations, and a color ccd camera for daytime. Several different algorithms will be used to provide robust tracking [8]. Some of the component systems, including Teleos AVP-100 and SENSAR PV-1, are commercially available while others are being developed locally at the Georgia Institute of Technology for use within this project. This paper describes how we are integrating these systems, taking advantage of their different tracking capabilities. Also presented in this paper are preliminary results using some of these methods.

2. Related Work

Extensive research has been conducted in the area of target tracking. Instead of attempting to provide an exhaustive reference, only representative work that is closely related to our objectives with the ASRT project is reviewed.

David et al [4] have present a real-time automatic target acquisition and tracking system in the context of dynamic battlefield environments. Their system, like ours, operates on natural environments which makes the tracking process much more difficult because of dynamic target obscurants like vegetation and dust. Their system uses real-time image processing hardware to track multiple tank-sized targets. They use a combination of best-first-search and probability-based algorithms for target tracking.

Samy [10] discusses different approaches to target tracking for natural textured backgrounds in real-time. He presents an overview of feature extraction and texture measures approaches. Other researchers, including Gilbert et al [5], and Hager et al [7] have developed real-time tracking systems using dedicated and non-dedicated hardware respectively.

In addition, Zheng and Chellapa [11] explore the detection of motion from imagery acquired from moving platforms. They address the difficulties created by the relative motion of the imaging device, and present a subpixel accuracy image registration algorithm to compensate for it. Their work estimates camera rotation and translations by matching feature points obtained using Gabor wavelet models. After the images are compensated for camera motion, they track the target using a motion-based approach.

3. ASRT Project Overview

The Autonomous Scout Rotorcraft Testbed (ASRT) project is a pilot project sponsored by the U.S. Army, with one of the two development efforts being conducted at the Georgia Institute of Technology. The goal of the overall project is to demonstrate autonomous scout rotorcraft operation of an unmanned air vehicle asset, providing an opportunity to send unmanned systems to look over/behind a hill and in the shadows of a hostile environment, and providing increased situational awareness for an aerial scouting team. The ASRT project is intended to promote new applications for robust flight control laws, integrated mission execution strategies, application of modeling and simulation, and vehicle robotics. In addition the program will define current difficult technical barriers, contribute to future full scale manned concepts with tests using unmanned flights of high risk concepts, and advance the state of the art for future autonomous vehicles. Other references to this project include [6, 3].

In this paper, we are primarily concerned with one of the subsystems within the ASRT project, the Mission Equipment Package (MEP). The objectives of the MEP include:

- Providing operator support in recognizing and designating a human-sized target,
- Autonomously tracking the target for periods of up to 30 minutes.
- Providing both day and night tracking capability.
- Displaying and recording real-time. stabilized day/night imaging from the helicopter on the ground station

In our system, we use the SENSAR PV-1 system to provide real-time image stabilization. This system consists of special-purpose hardware that supports pyramid-based algorithms. The use of pyramid structures in image processing and tracking has been presented by Burt [1], and Clark [2] among others.

4. Tracking Strategies

Our tracking strategy is based on the combination of several component subsystems. These include SENSAR's PV-1, Teleos AVP-100, and an SGI Indy workstation running other motion-based tracking algorithms.

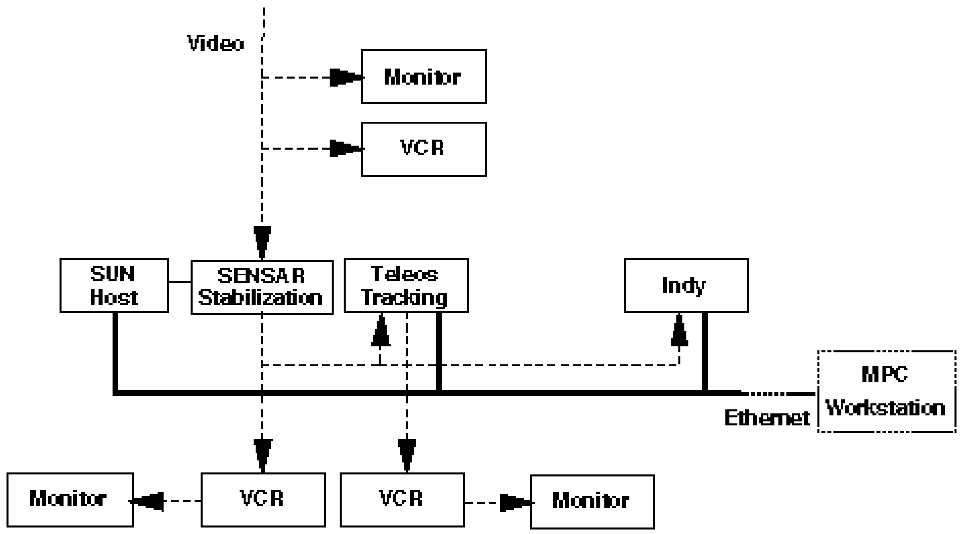

Figure 1: MEP Ground Station Architecture

Figure 1 shows how these systems work together on the ground station. Video coming from the air vehicle via the video link is sent as input to the PV-1 system which provides real-time image stabilization. After the target has been designated by the operator, the stabilized video is then sent to the Teleos AVP-100 and to the Indy workstation for tracking. The output from these tracking systems is displayed and archived as necessary.

As mentioned above, the role of the PV-1 system is to provide real-time stabilization of the video signal comming from the vehicle. This system posses dedicated special-purpose hardware to construct image pyramids. The image alignment process computes optical flow using local correlation estimates on these structures and then constructs a parametric model to minimize the difference between images [9].

The Teleos AVP-100 is real-time tracking system which has the capability to produce motion measurements within a subimage that is selected under software control. The user can select the subimage size and dynamically control the region of interest. The development of a custom tracking algorithm using this system is currently underway.

As mentioned before, a motion-based tracking algorithm will be running on the SGI Indy workstation. Two different motion algorithms have already been developed and tested using imagery from the demonstration test site at Berry College in North Georgia that has been stabilized using the PV-1 system. Both of these algorithms were developed during earlier stages of the project. The goal of the first method is to track the target in real-time; the second one is concerned with extraction of motion-based objects. Each of these algorithms is presented, along with preliminary results, in the remainder of this paper.

4.1 Method One: A Real-Time Tracking Algorithm

As we mentioned previously, the goal when developing this algorithm was to achieve real-time target tracking using a non-dedicated workstation. This requirement severely reduces the amount of computation that can be performed between two consecutive frames. In order to meet this constraint two commitments were made. First, only local processing of the each window was conducted (sub-windowing). Second, complex feature extraction or segmentation of the image was avoided.

Using this approach, the operator can dynamically locate the center of this region anywhere within the image during the target designation phase. The actual size of the subimage is variable and changes according to the amount of motion detected within it. Basically, the algorithm simply computes image differences between two consecutive frames and then tracks the centroid of the difference from frame to frame. The size of the region of interest varies along both dimensions according to the respective variances over the computed image difference.

As customary, the image difference between two frames f, g over a given region is defined as

$$d(i,j) = \mid f(i,j) - g(i,j) \mid$$

where

$$\mathrm{x} - \lfloor \tfrac{n}{2} \rfloor \leq \mathrm{i} \leq \mathrm{x} + \lfloor \tfrac{n}{2} \rfloor,\ \mathrm{y} - \lfloor \tfrac{m}{2} \rfloor \leq \mathrm{j} \leq \mathrm{y} + \lfloor \tfrac{m}{2} \rfloor$$

and the window is centered at x, y.

We further define

$$d'(i,j) = \begin{cases} 1, & d(i,j) > \epsilon \\ 0, & otherwise \end{cases}$$

where ϵ is some threshold. This is done to eliminate some of the noise that is always present in images and to obtain a binary image.

Next, we compute the centroid of the window. This is the pair $(\bar{x}, \bar{y})$ where

$$\bar{x} = \frac{1}{nm} \sum_{i=x-\lfloor\frac{n}{2}\rfloor}^{x+\lfloor\frac{n}{2}\rfloor} \sum_{j=y-\lfloor\frac{m}{2}\rfloor}^{y+\lfloor\frac{m}{2}\rfloor} d'(i,j) \cdot i, \quad \bar{y} = \frac{1}{nm} \sum_{i=x-\lfloor\frac{n}{2}\rfloor}^{x+\lfloor\frac{n}{2}\rfloor} \sum_{j=y-\lfloor\frac{m}{2}\rfloor}^{y+\lfloor\frac{m}{2}\rfloor} d'(i,j) \cdot j$$

The centroid is interpreted to be an estimate of the center of the moving object, and the center of the window is repositioned to be at $(\bar{x}, \bar{y})$ for the next iteration of the algorithm. Also, the variance along each dimension is computed. These are defined as

$$x_\sigma = \sqrt{\left(\frac{1}{nm} \sum_{i=x-\lfloor\frac{n}{2}\rfloor}^{x+\lfloor\frac{n}{2}\rfloor} \sum_{j=y-\lfloor\frac{m}{2}\rfloor}^{y+\lfloor\frac{m}{2}\rfloor} d'(i,j) \cdot i^2\right) - \bar{x}^2},$$

$$y_\sigma = \sqrt{\left(\frac{1}{nm} \sum_{i=x-\lfloor\frac{n}{2}\rfloor}^{x+\lfloor\frac{n}{2}\rfloor} \sum_{j=y-\lfloor\frac{m}{2}\rfloor}^{y+\lfloor\frac{m}{2}\rfloor} d'(i,j) \cdot j^2\right) - \bar{y}^2}$$

The size of the window along the x-axis for the next iteration is set to be $1.5\, x_\sigma$, and similarly the size along the y-axis is set to be $1.5\, y_\sigma$. This is done so that the size of the window will increase or decrease according to the amount of motion detected within it. The size of the window along either dimension is restricted to be within a range specified by the user (in our implementation the minimum along either dimension was set to be 10 pixels and the maximum 70). This allows the user to limit the minimum and maximum size of the tracking window.

If the number of non-zero valued pixels inside the window falls below some threshold (10 pixels is the default), the target is declared lost and the window is placed at the center of the image with maximum size along both dimensions.

4.2 Method Two: Extracting Motion-based Objects

As noted earlier the previous algorithm does not recognize or identify its targets; it only follows movement. We now describe an algorithm developed by Zheng and Chellapa [11] used for this method. It starts from the observation that more can be done than just thresholding to reduce noise. The key point is that if a point in an image is moving, then it is very likely that some or all of its neighbors are moving too. Thus we can discard isolated points in the difference image. They also point out that the threshold value should not be a constant. Instead it should vary accordingly to the local contrast of the image. As the contrast increases, the threshold should also increase. Thus they classify a point as a moving point if

$$min\{\overline{d(i,j)}, d(i,j)\} > max\{\gamma D(i,j), \epsilon\}$$

where

$$\overline{d(i,j)} = \frac{1}{N_\omega} \sum_{i,j \in \omega} d(i,j),$$

$$D(i,j) = sup_{i,j\in\omega}\{f(i,j)\} - inf_{i,j\in\omega}\{g(i,j)\},$$

γ is the relative threshold
ω is the neighborhood of (i,j)

In other words, a given point is punished for having neighbors with low values in the difference image and for being in a high contrast area.

Their algorithm tries to detect the moving object on each image. They let Ω be the points of the object in a given image. Thus if we consider three consecutive images f_1, f_2 and f_3, then we could detect $S_1 = \Omega_1 \cup \Omega_2$ by means of the method just described. There is a problem though, since points that belong to an object tend to have similar intensities, at least locally. In practice S_1 is not detected but rather a subset of it. The subset not usually detectable is the one where the points that happened to move to a location that was previously occupied by a point with the same or very similar intensity value. Thus formally, using the previously described method to detected moving points, the set detected is

$$\bar{S_1} = S_1 - S_1{}'$$

where the displacement from Ω_1 to Ω_2 is assumed to be (u_1, v_1) and

$$S_1{}' = \{(i,j) \mid (i,j) \in \Omega_1 \cap \Omega_2 \ \ and \ \ f_1(i,j) = f_1(i - u_1, j - v_1)\}$$

Similarly

$$\bar{S_2} = S_2 - S_2{}'$$

and

$$S_2{}' = \{(i,j) \mid (i,j) \in \Omega_2 \cap \Omega_3 \ \ and \ \ f_2(i,j) = f_2(i - u_2, j - v_2)\}$$

where we assume that the displacement from Ω_2 to Ω_3 is (u_2, v_2)

It is easy to see that if $(u_1, v_1) = (u_2, v_2) =$ (u, v) then

$$(i,j) \in S_1' \Longleftrightarrow (i + u, j + v) \in S_2'$$

and that

$$(\bar{x_2}, \bar{y_2}) = (\bar{x_1} + u, \bar{y_1} + v)$$

where $(\bar{x_2}, \bar{y_2}), (\bar{x_1}, \bar{y_1})$ are the centers of mass of f_1 and f_2.

Now, that we have all the definitions we can go back to the algorithm. The first thing to do is to compute $\overline{S_1}$. Next, estimate Ω_1 by taking the closure of $\overline{S_1}$. The closure is constructed recursively from $\overline{S_1}$ by connecting any two points in it that have a common coordinate. Once the closure has been computed, translate f_1 by (u,v) calling it f_1', and compute its difference with f_2. Finally, a point (i,j) in the translated closure is determined to be a point of the moving object in image f_1 if its value in the new difference image is below some threshold, i.e., $\mid f_1'(i,j) - f_2(i,j) \mid \leq \epsilon$ and either it belongs to $\overline{S_1}$ or at least one of its 3x3 neighbors is in Ω_1.

In other words, what this algorithm does is the following. It computes the difference between f_1 and f_2, and attempts to filter out as much noise as possible. Then it computes the closure of this difference to try to make up for the holes due to similar gray level of moving points. Next it computes the difference between f_2 and f_3, to estimate the displacement from f_1 to f_2. Then it shifts f_1 by that displacement, and finally looks for points in the closure that yield zeros in the new difference image.

The assumptions that this algorithms make about the object, more precisely the motion of its center of mass and the relative position among points on it, suggest that this algorithm should work reasonably well with rigid bodies, but if the object being tracked is non-rigid, then the assumptions made do not hold any longer. The assumption that the displacement from Ω_1 to Ω_2 is (u_1, v_1) and $S_1{}' = \{(i,j) \mid (i,j) \in \Omega_1 \cap \Omega_2 \ \ and \ \ f_1(i,j) = f_1(i - u_1, j - v_1)\}$ is equivalent to saying that we are dealing with one rigid body in purely translational motion, or with more than one rigid body all with the same translational motion. This algorithm was implemented to determine its performance when the object being tracked is a human walking (a deformable, non-rigid object), and thus its assumptions do not strictly hold.

5. Preliminary Results

Both algorithms have been run on video imagery of a human walking in outdoors environments, including the demonstration site chosen for the ASRT Project. The video was stabilized using the SENSAR system before being fed to the algorithms which were run on a non-dedicated SGI Indy Workstation. The results obtained are presented below.

5.1 Method One

This real-time algorithm worked well as long as the SENSAR system could provide stabilized imagery without introducing considerable translations from one frame to the next (i.e., saccadic motion). In addition, the algorithm worked best when the area occupied by the target was neither too large nor too small. If the area was too big, about 1/3 or more of the pixels on each dimension, the high frequency noise introduced by the moving grass and leaves caused the algorithm to lose the target. On the other hand, the algorithm could not extract the target from the background if its size was too small. This occurred when the target size was less than 1/10 of the total number of pixels on each dimension. Figures 2 and 3 show typical results obtained using this algorithm.

Figure 2: Sequence obtained using real-time tracking algorithm.
This sequence illustrates the performance of the algorithm against a relatively static background. It proceeds in time from right to left and from top to bottom. The detected target is shown within the box.

5.2 Method Two

The current implementation of this motion-based object tracking algorithm on the SGI Indy workstation, does not run in real-time. The user can select the position to start tracking from within the image, and then the algorithm collects a sequence of frames (six in our implementation), and then processes them. The processing is further limited to a subwindow within the image (30 pixels along both dimensions). In these experiments, it was observed that the algorithm tends to detect parts of the target, even though sometimes it also selects sections of the environment. This behavior, however, can be easily understood from the description of the algorithm, since it assumes that there is only a single moving object. Figures 4 and 5 show typical results obtained using this algorithm.

6. Summary

This paper represents an early report on the visual target designation and tracking capabilities for an autonomous helicopter. The basic components of the Mission Equipment Package tracking system for the ASRT project have been described, including an overview of the interconnections between them and the data flow among them. In addition, preliminary results were described and presented that were obtained using two motion-based tracking algorithms operating on stabilized imagery of a human walking in natural environments. It should be reiterated that these are early results in a project which will be tested in the field in the Spring of 1996. Future publications will describe the evolution of this system as new modules such as the Teleos system are integrated.

Acknowledgments

This research is funded by the U.S. Army AATCOM at Fort Eustis, VA under Contract #DAAJ02-94-C-0021. Additional funding for the Mobile Robot Laboratory is provided by ARPA, the Office of Naval Research, and the National Science Foundation.

References

1. Burt, P., "Smart Sensing within A Pyramid Vision Machine", *Proceeding of the IEEE*, Vol. 76, No. 8, pp. 106-144, August 1988.
2. Clark, M., "Robot-based real-time motion tracker", *Proc. SPIE Symposium on Advances in Intelligent Robotics Systems*, pp. 1-7, 1989.
3. Collins, T., Cardoze, D., and Gerber, D., "Object-Oriented Development of an Integrated Processor System for an Unammed Aerial Vehicle", *Proc. AUVS '95 Symposium*, Washington DC, July 1995.
4. David, P.,Balakirsky, S., and Hillis, D., "A Real-Time Automatic Target Acquisition System", *Proc. AUVS '90 Symposium*, Dayton, OH, pp 1-16, July-Aug. 1990.
5. Gilbert, A., , Giles, M., Flachs, G., Rogers,R., Hsun, Y., "A Real-Time Video Tracking System", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol PAM1-2, No 1, pp 47-56, January 1980.
6. Gordon, M. and Schrage, D., "Development Of The Georgia Tech Autonomous Scout Rotorcraft Testbed", *Proc. AUVS '95 Symposium*, Washington D.C., July 1995.
7. Hager, G., Toyama, K., "A Framework for Real-Time Window-Based Tacking Using Off-The-Shelf Hardware", DRAFT Documentation For Version 0.95 alpha, Yale University, Department of Computer Science, October 16, pp 1-10, 1994.
8. Kluge, K. and Thorpe, T., "Explicit Models for Robot Road Following", *Proc. 1989 IEEE International Conference on Robotics and Automation,* May 1989, Scottsdale, AZ, pp. 1148-54.
9. Pyramid Vision 1 Alignment/Disparity Library User's Guide, SENSAR Inc, 1994.
10. Samy, R., "Real-Time Tracking of Target Moving on Natural Textured Background", 'it SPIE Vol. 595 Computer Vision for Robots, pp. 142-150, 1985.
11. Zheng, Q. and Chellapa, R., "Motion Detection In Image Sequences Acquired From A Moving Platform", *Proc. IEEE International Conference On Acoustics Speech and Signal Processing*, Minneapolis, MN, April 27-30, pp 201-204, 1993.

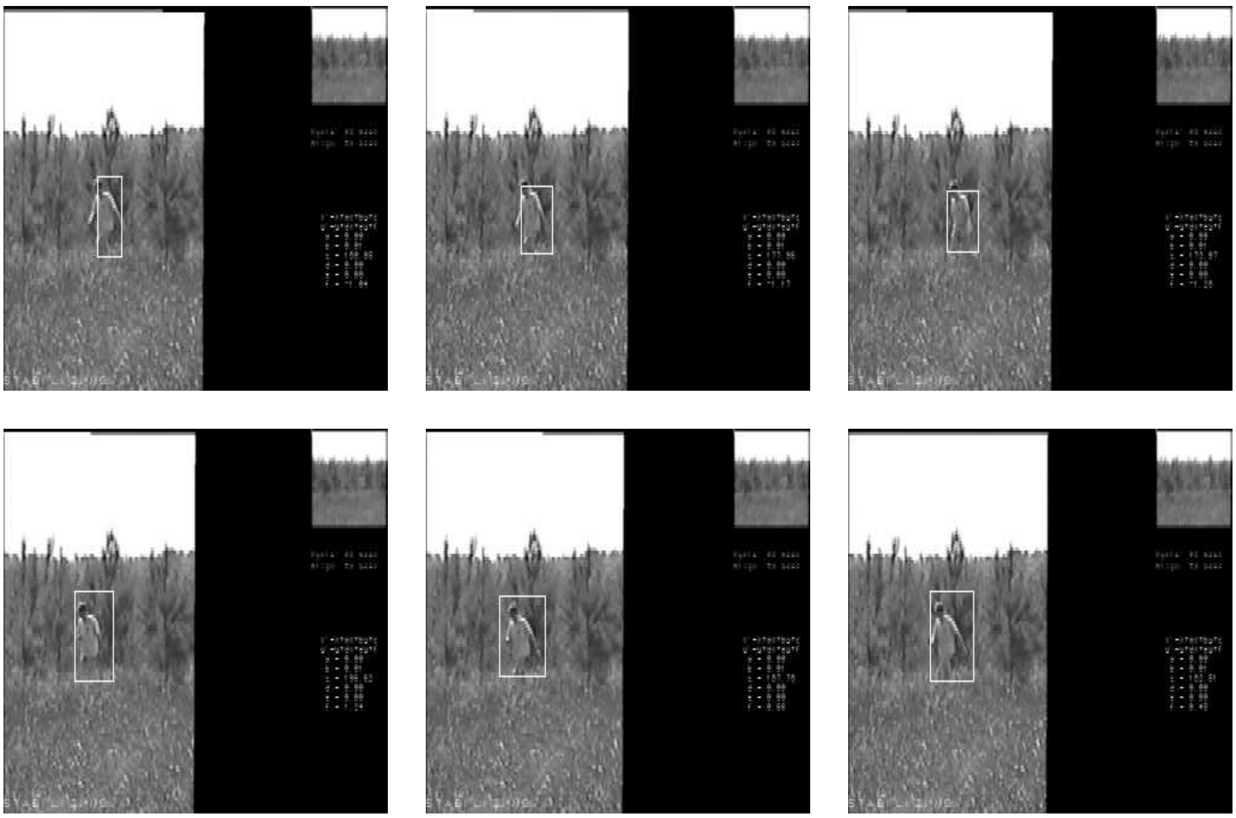

Figure 3: Another sequence obtained using real-time tracking algorithm.
This sequence taken at the Berry College demonstration site illustrates the algorithm's performance against obscurants. The sequence proceeds in time form right to left and from top to bottom. Notice how the size of the interest area defined by the white rectangle changes according to the size of the target in the frame.

Figure 4: Sequence obtained using Zheng-Chellapa algorithm.
This sequence, also taken at the demo site, proceeds in time from top to right to left and from top to bottom. Notice how features are not extracted sometimes due to high similarity between frames. Occassionally features from the background get extracted as well.

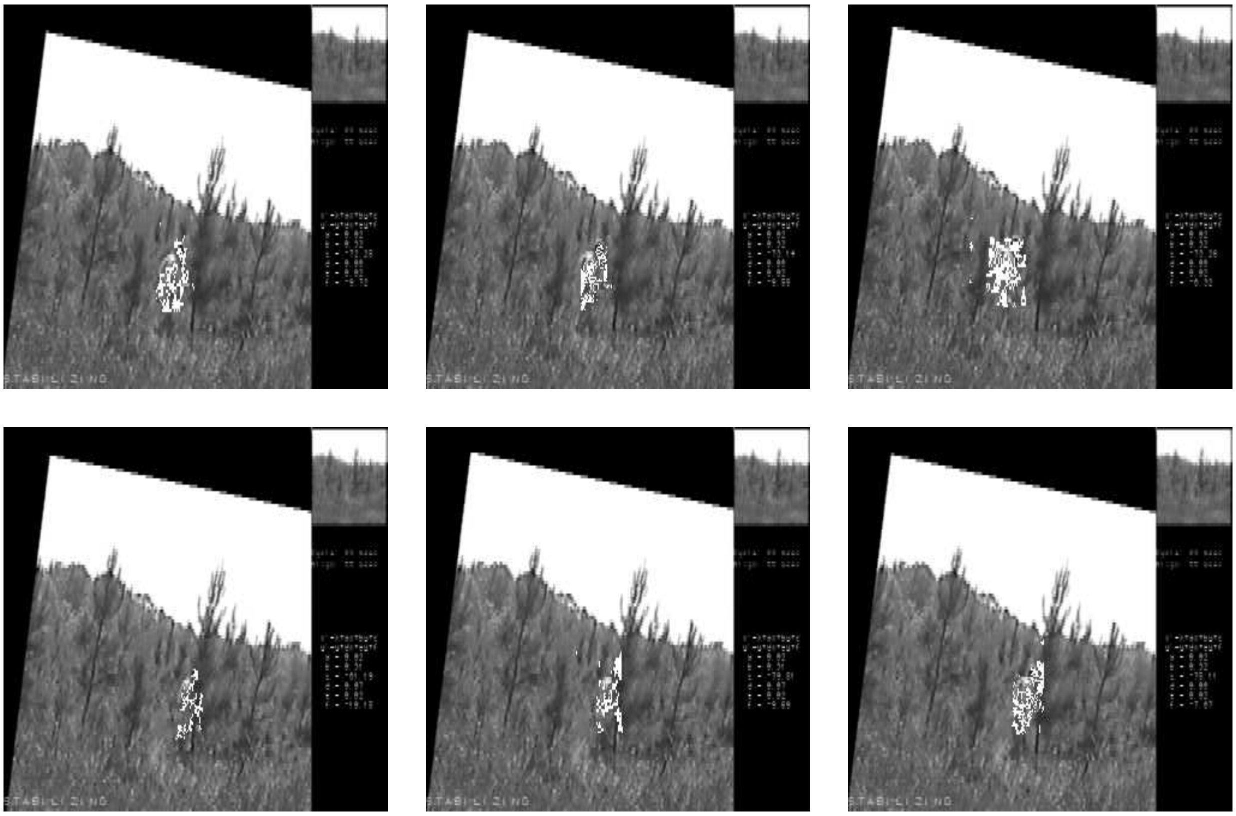

Figure 5: Another sequence obtained using Zheng-Chellapa algorithm.
This sequence, also from the demo site, illustrates the performance of our implementation agains obscurants. Notice how regions of the background environment are selected by the algorithm due to their constant motion.