World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:7, No:1, 2013

A Hybrid CamShift & l_1-Minimization Video Tracking Algorithm

Clark Van Dam, and Gagan Mirchandani

International Science Index, Computer and Information Engineering Vol:7, No:1, 2013 waset.org/Publication/1686

***Abstract*—**The Continuously Adaptive Mean-Shift (CamShift) algorithm, incorporating scene depth information is combined with the l_1-minimization sparse representation based method to form a hybrid kernel and state space-based tracking algorithm. We take advantage of the increased efficiency of the former with the robustness to occlusion property of the latter. A simple interchange scheme transfers control between algorithms based upon drift and occlusion likelihood. It is quantified by the projection of target candidates onto a depth map of the 2D scene obtained with a low cost stereo vision webcam. Results are improved tracking in terms of drift over each algorithm individually, in a challenging practical outdoor multiple occlusion test case.

***Keywords*—**CamShift, l_1-minimization, particle filter, stereo vision, video tracking.

I. Introduction

THE CamShift tracking algorithm, an adaptation of the Mean Shift algorithm, is a well established iterative kernel-based algorithm known for its efficiency and effectiveness in tracking targets in a sequence of video frames [1]. It works by matching the color histogram of the target to an image patch in each frame. However, it is well recognized that because features tracked are color based, it does not perform well under severe illumination changes, occlusion, or when multiple objects or portions of the background have the same color histogram as the target [2].

Improvement and variation of the CamShift tracking algorithm is an active area of research. Recent work includes combination with particle filters [3], motion segmentation [4], silhouette detection [5] and feature matching [6]. The work presented here builds on the novel idea of combining the two major classic methods of tracking, namely kernel-based and state space-based methods, but using recent extensions of both. We employ depth information using stereo vision with CamShift, as recently proposed in [7] and for the state-space approach, utilize the new development of application of sparse sampling to tracking with particle filtering [8].

Within a typical tracking sequence there are invariably portions where one or the other algorithm is better suited. We employ a hybrid scheme which is seen to outperform either algorithm on its own. The main contribution in this work is the combination of the CamShift and l_1-minimization algorithms in a particle filter framework, to create a robust-to-occlusion hybrid tracker that takes advantage of the efficiency of the former and effectiveness of the latter in adapting to appearance changes of the target. The novelty of the algorithm is in adopting a hybrid scheme and in using depth measurements for drift and occlusion detection, in order to transfer control from one tracker to the other. The interchange scheme operates by projecting the target candidates in the 2D frame sequences onto a depth map generated separately for the static field of view. The depth map, covering the background of the same field of view as the 2D camera, need only be measured once for a fixed camera position. Stereo vision, utilizing a low cost 3D webcam, was used to generate the depth map. Other depth sensors could also be used.

C. Van Dam and G. Mirchandani are with the Signal Processing Group, University of Vermont, Burlington, VT 05405 USA (e-mail: cvandam@uvm.edu; gmirchan@uvm.edu).

II. Background

A. CamShift

The CamShift algorithm essentially adaptively updates the Mean-Shift (MS) algorithm target candidate window size. Since the region being tracked can change in location, appearance and size between frames, the MS algorithm is used as a way to converge from an initial guess for location and scale to the best match based on the color-histogram similarity. Note that these algorithms are more general, and can track any kind of feature, but color is the most popular [9].

In general, for a given image (or a sequence of images), spatial locations, color, texture, motion or parameters of curves or surfaces, extracted from local neighborhoods, can be mapped into a space. Significant features then correspond to high density regions in the space. That is, n observations $\{x_i\}_{1\ldots n}$ in *d*-dimensional Euclidean space $\Re^d$ can be seen to be drawn from an unknown pdf f. While the feature space may have irregular data clusters, it is seen that the *modes* (local maxima) of the underlying f provides a reliable method for determining the cluster centers of the features [10]. For low to medium data sets, estimating the pdf $\hat{f}$ through the kernel estimation method appears a good practical choice (ibid.).

For real time tracking of non-rigid objects, MS iterations are used to find the most probable target position in the current frame. Essentially, the MS algorithm is an iterative, gradient ascent method for finding the local maximum of a target kernel-density distribution for the purposes of region matching. The match criterion is similarity based on a color histogram.

B. l_1-minimization

The l_1-minimization tracking algorithm [8], [11] is a recent

International Science Index, Computer and Information Engineering Vol:7, No:1, 2013 waset.org/Publication/1686

and sophisticated innovation that casts the tracking problem in the framework of compressive sampling or what is also known as compressed sensing [12], [13]. The latter, which has generated considerable excitement and a plethora of publications and research activity seems to have far reaching implications. The advantage of the l_1-minimization tracker is that it is robust to occlusion, and adaptively generates templates that evolve as the target undergoes affine distortion (scale, rotation, translation, shear). The disadvantage is in terms of computational requirements unsuitable for real-time implementation.

The algorithm entails representing a target candidate as a weighted sum of a *template set* and a *trivial template set*, referred to collectively as basis templates. The template set can be obtained from an initial selection, reference databases, or more generally from the target appearance in previous frames. Trivial templates are single pixel perturbations added to represent occlusion of the target candidates. The l_1-minimization problem requires the determination of the sparsest representation of the target candidate in terms of the weighted basis templates. Tracking continues in a particle filter framework that serves to propagate sample distributions over time.

Hence, the l_1-minimization video tracking problem proceeds as follows: Find a sparse representation in a template subspace. Solve a l_1-regularized least squares problem to achieve a sparse representation. Use sequential state estimation to track the desired object, utilizing a particle filter framework to recursively reconstruct the a posteriori pdf of the state.

III. Method

A. Threshold Parameters and Transfer of Control

The key problem to formulating a hybrid algorithm relates to the criteria of interchanging control from one algorithm to the other. How does either tracker know whether it is tracking the right target? Generally it does not. All trackers are subject to drift, but some simple criteria can be set to assess the status of the tracking state. Drift occurs when the bounding box forming the target candidate tracks the wrong target, which is especially likely to occur when the target becomes occluded. Occlusion tends to introduce excessive target candidate size and velocity variation and is detected by differences in depth measurements across the span of a partially occluded target candidate.

The thresholds for control transfer are based on absolute and relative parameters related to changes in target size and position. Measurements of changes in target size and position in physical units are related to appearance in terms of pixels. In that way, static thresholds on the physical size and velocity of the target are translated into dynamic thresholds in terms of pixels, depending on the target appearance from the geometric perspective in relation to the camera. The threshold parameters are summarized as follows:

- Absolute Target Size in Pixels
  - Maximum (w_{max}) and minimum (w_{min}) width
  - Maximum (h_{max}) and minimum (h_{min}) height
- Absolute Target Position in Pixels
  - Proximity to image border, for out-of-frame detection.
- Relative Target Velocity in Pixels
  - Maximum horizontal $\left(v_{x_{\max}} = \Delta x_{\max}\right)$, vertical $\left(v_{y_{\max}} = \Delta y_{\max}\right)$, width $\left(v_{w_{\max}} = \Delta w_{\max}\right)$ and height $\left(v_{h_{\max}} = \Delta h_{\max}\right)$ velocities.
  - Maximum change in $w*h$ product.
- Relative Target Acceleration in Pixels
  - Maximum horizontal $\left(a_{x_{\max}} = \Delta v_{x_{\max}}\right)$, vertical $\left(a_{y_{\max}} = \Delta v_{y_{\max}}\right)$, width $\left(a_{w_{\max}} = \Delta v_{w_{\max}}\right)$ and height $\left(a_{h_{\max}} = \Delta v_{h_{\max}}\right)$ accelerations.
- Change in Distance (z) to Target in Physical Units
  - Used to set parameters for Camshift based on near, mid, and far field distance to target. This affects the size and location of the target candidates generated, as opposed to the other thresholds, which quantify the drift or occlusion likelihood of the candidates provided.
  - Obtained as average distance associated with pixels of the target candidate.
- Size of Target in Physical Units
  - Maximum width and height as percentage of difference between extreme depth values.
  - Obtained by averaging the distance in the depth values associated with opposite ends of the target candidate.

These parameters set simple thresholds to assess drift and occlusion likelihood, and are used to pass control between CamShift and the l_1-minimization tracker.

B. Depth Map

Stereo vision was used to generate the depth map, but other methods such as LIDAR, RADAR, or even human estimation of the scene depth could be used. The stereo camera was first calibrated to account for optical misalignments. Then a left and right stereo image pair was obtained of the background scene (no moving objects), each overlapping the region of interest in the 2D camera setup. The image pair was rectified for stereo matching. The disparity was used to generate a depth map where for each (x,y) pixel, a grayscale value is taken as the distance, to within a constant, of the physical distance to the object represented by the pixel. Proper choice of this constant calibrates the depth measurements. Maximum and minimum depth values were clamp limited between 0 and 255 (256 grayscales), respectively, as appropriate for the desired contrast of the scene. Lastly, the depth map needed to be transformed to the 2D camera scene, positioned to overlap the portion in common.

C. Relationship of Pixel and Physical Dynamics

The CamShift tracker is first initialized with the depth map, initial horizontal and vertical pixel coordinates $x(t)\big|_{t=0}$, $y(t)\big|_{t=0}$, frame rate (e.g. 30fps), frame dimensions (e.g.

International Science Index, Computer and Information Engineering Vol:7, No:1, 2013 waset.org/Publication/1686

640x480) and initial target pixel width and height $w(t)\big|_{t=0}$, $h(t)\big|_{t=0}$.

In terms of pixels, the 2-dimensional kinematics between each consecutive frames are described by (1) and (2), relating position (x, y), velocity (v_x, v_y), and acceleration (a_x, a_y) components.

$$\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}=\begin{pmatrix}x-x_0\\ y-y_0\end{pmatrix}=\begin{pmatrix}v_x t+\frac{1}{2}a_x t^2\\ v_y t+\frac{1}{2}a_y t^2\end{pmatrix}. \quad (1)$$

$$\begin{pmatrix}\Delta v_x\\ \Delta v_y\end{pmatrix}=\begin{pmatrix}v_x-v_{x_0}\\ v_y-v_{y_0}\end{pmatrix}=\begin{pmatrix}a_x t\\ a_y t\end{pmatrix}. \quad (2)$$

For each pixel, the depth map associates a physical distance $z'(x, y)$, where the prime " ' " is used to indicate a quantity in physical units (i.e. m, $\frac{m}{s}$, $\frac{m}{s^2}$). Unprimed quantities are in terms of pixels (i.e. pix, $\frac{pix}{frame}$, $\frac{pix}{frame^2}$). Corresponding to (1), for the 3rd-dimension is (3). A 3rd-dimensional correspondence to (2) in primed coordinates is inaccurate unless a very accurate depth map is available, the frame rate is very high, and the resolution is large. That is, it is difficult to measure acceleration accurately.

$$(\Delta z')=(z'-z'_0)=\left(v'_z t+\frac{1}{2}a'_z t^2\right). \quad (3)$$

The depth map gives a measure of the third spatial dimension, in physical units, which is projected onto the 2D pixel grid in order to estimate the relationship between pixel distances and pixel velocities to physical units. The key projection which relates pixel velocities to velocities in physical units is given in (4) and illustrated in Fig. 1.

$$\begin{pmatrix}v'_x\\ v'_y\end{pmatrix}=\begin{pmatrix}\Delta x\dfrac{v'_z}{\sqrt{\Delta^2 x+\Delta^2 y}}\\ \Delta y\dfrac{v'_z}{\sqrt{\Delta^2 x+\Delta^2 y}}\end{pmatrix}. \quad (4)$$

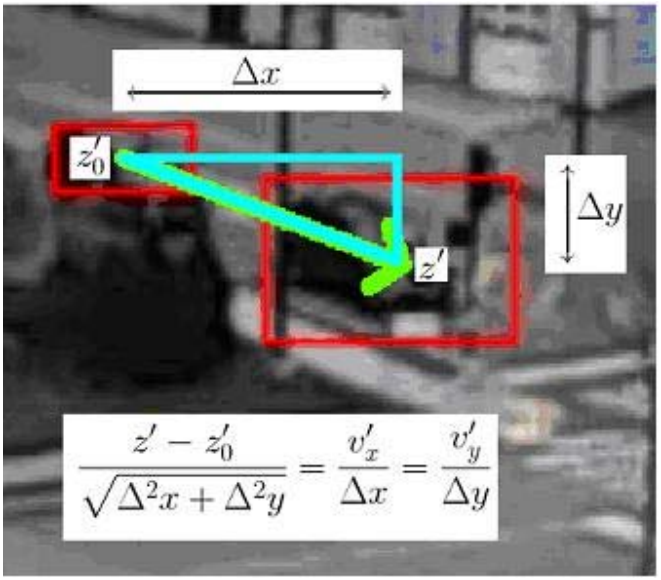


Fig. 1 Illustration of projection in (4). The initial and final target coordinates, bounded in boxes, each have an associated depth distance, the change of which is used to relate pixel distance to physical distance.

As a simplifying assumption, physical acceleration components in (1)-(3) are assumed zero. The frame-to-frame change in vertical and horizontal position in terms of physical distance is calculated with the physical velocities obtained using (4), and the time between frames, $t=\frac{1}{30[fps]}$, shown explicitly in (5).

$$\begin{pmatrix}\Delta x'\\ \Delta y'\end{pmatrix}=\begin{pmatrix}v'_x\dfrac{1}{30[fps]}\\ v'_y\dfrac{1}{30[fps]}\end{pmatrix}. \quad (5)$$

Now the key relationship between physical distances and pixel distances is obtained as the ratio $scale=\left|\frac{\Delta x}{\Delta x'}\right|=\left|\frac{\Delta y}{\Delta y'}\right|$, which dynamically varies with each frame as the target is projected across varying depth. This ratio is used to obtain pixel thresholds by multiplying it by measurements of physical position and size components.

Physical measurements of average length and width are limited. It is also useful to impose a target size limitation strictly in terms of pixels, such that target length and width changes are limited to a reduction by 50% or enlargement by 200% frame-to-frame. In addition, fixed limits on physical velocities $v'_{x_{\max}}, v'_{y_{\max}}, v'_{w_{\max}}$, and $v'_{h_{\max}}$ are translated in terms of pixels, and altogether the most restrictive limit sets effective $\Delta x_{\max}, \Delta y_{\max}, \Delta w_{\max}$, and $\Delta h_{\max}$ velocities.

Targets directly moving nearly in-line with the eye of the camera present a challenge, as a large $\Delta z'$, in terms of

International Science Index, Computer and Information Engineering Vol:7, No:1, 2013 waset.org/Publication/1686

physical distance, may be projected onto small Δx and Δy pixel changes, and used to infer an erroneously large $\Delta x'$ and $\Delta y'$. In such cases, bounds on width *w* and height *h* changes (scale velocities and accelerations), generated analogous to (1) and (2), dominate the threshold conditions. Generally limits on *x*, *y*, *w*, *h* change and rate-of-change are imposed at any instant, but *x* and *y* limits dominate when motion is equidistant to camera and *w* and *h* limits dominate for motion in the *z'*-direction.

The key idea is to *approximately* and *locally* relate pixel distances between the target location in consecutive frames to physical units of target velocity and acceleration. Then measurements of the physical distance and size of the target can be translated into pixel bounds specific to the geometric relationship between the camera and target, which vary frame-to-frame. This method allows for rapid scale variation in target appearance if the target is on the terrain close to the camera, but only minimal scale variation if in the far-off distance. Similarly, larger velocity and acceleration thresholds are permitted for close up objects.

D. Hybrid Tracking

Tracking starts out using the CamShift algorithm. When occlusion or drift is detected, the target size and location from several frames back, before CamShift was deemed to fail, is used to initialize l_1-minimization tracking. Typically one to ten frames back are used, depending on how tight the other parameters are set. The CamShift algorithm will typically show variations approaching threshold conditions just prior to any of the thresholds being met, so it is thought that the tracking results from several frames back are generally better estimates. However, tight thresholds can also be used and the very last frame meeting them used to initiate l_1-tracking. Loose thresholds will cause the tracker to drift from the true target and latch onto the background or another moving object. Tight thresholds break the algorithm when the thresholds fall outside the bounding ranges. This is the trade off between tracking accuracy and the computational expense of invoking l_1.

The l_1-tracker takes the target coordinates generated from CamShift and initializes a set of ten target templates based on it and perturbations thereof [8]. Tracking continues by l_1 minimization and particle filtering. It takes as many frames as target templates (ten) for the l_1-tracker to build up a unique and representative set of target templates. Hence it is allowed to do so. Periodically thereafter the tracking results of the l_1 tracker are used to try to reinitiate CamShift tracking. If CamShift is able to continue tracking for a set number of frames (e.g. ten) without exceeding thresholds, then the l_1 tracker is shutdown and tracking resumed again by CamShift. The measure of the target size allows for criteria to detect full occlusion and ensure control is not passed back to CamShift while the target is still occluded. The process repeats, interchanging control between algorithms. The idea is that CamShift only calls upon the l_1-tracker when needed, and interchange of control is based on occlusion likelihood.

IV. Experimental Setup

Tracking sequences were obtained utilizing a standard 2D 640x480 pixel RGB camera at 30fps. The depth map was obtained by processing a pair of stereo frames from a low cost Minoru 3D Webcam, positioned to overlap a portion of the field of view from the 2D camera. The CamShift and control algorithms were implemented in C++ utilizing OpenCV libraries running on Linux platform AMD quad-core laptop. The l_1-minimization tracker is implemented in MATLAB, cooperating by sharing a file space in which to exchange status flags and other input/output parameters. The source code and experimental results are posted at http://www.cems.uvm.edu/~gmirchan/software/caml1/.

V. Experimental Results

A practical multiple occlusion test case was chosen to illustrate the hybrid tracker performance and that compared to the performance of the component algorithms which make it up. The depth map, shown in Fig. 2 overlapping a portion of the 2D camera field-of-view, was measured from stereo image pairs acquired during full summer bloom, in order to highlight the tree occlusions. The noise in the depth map introduced uncertainty into distance measurements, and made acceleration estimates wholly unreliable, so the acceleration components in (1)-(3) were set to zero. Even so, the target size measurements were stable enough to set thresholds as criteria for occlusion detection.

The target is a car coming to a stop and making a turn, occluded by a lampost, traffic control post, and two trees, one after another. Neither the l_1-tracker or CamShift on its own were able to track the target in this sequence. The results using [8] implementation of the l_1-tracker are shown in Fig. 3. It drifts off the target after 123 frames due to the series of occlusions disrupting the template set, including the branches of a bare tree clouding the intersection. As the car passes the last tree occlusion, the template set is disrupted to the point that the target is lost in Fig. 3 (d).

The sequence of results using CamShift only are shown at the beginning of the hybrid sequence in Fig. 5 (a)-(d). The target height in pixels rapidly increases, and shortly after Fig. 5 (d), the target candidate drifts onto the traffic control post at Frame 01892.

International Science Index, Computer and Information Engineering Vol:7, No:1, 2013 waset.org/Publication/1686

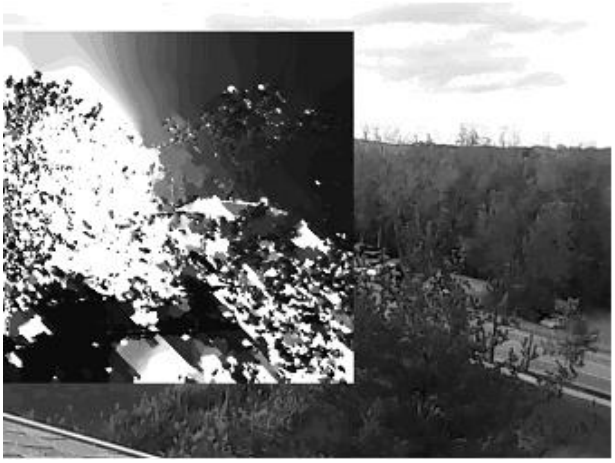


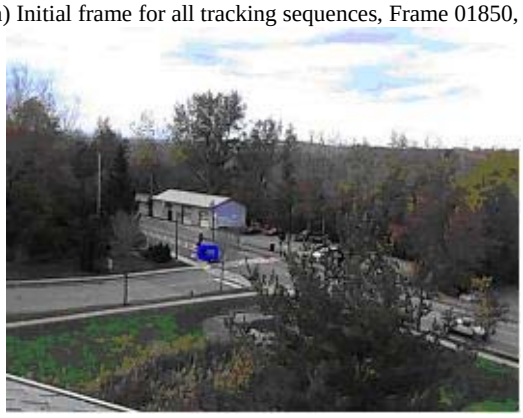
Fig. 2 Overlay of depth map in field of view of 2D frames, in the region of interest where target is tracked. White pixels appear as near to the camera, whereas black pixels appear as a far distance. The foliage is highlighted in part due to the non-simultaneous acquisition of the left and right stereo pairs from the webcam, in between which the foliage blows in the wind. Using a larger stereo baseline and post processing techniques would improve the quality of the depth map and hybrid algorithm performance

Fig. 5 shows the successful hybrid tracking results for the video sequence. CamShift is deemed to fail when it hits a threshold. Control is then transferred to l_1 using the target coordinates generated by CamShift ten frames prior to failing, since that is deemed a reliable tracking result. The control transfer from l_1 back to CamShift is attempted periodically every ten frames, and control is actually handed over if CamShift can continue tracking for at least ten frames on reinitialization without exceeding threshold limits on size and velocity. The number of frames involved in each part of the transfer process are parameters that can be freely set.

Hybrid tracking starts out by CamShift, as shown in Fig. 5 (a)-(d) until tracking drifts in Frame 01892, due to occlusions in the intersection. This occlusion is detected when the frame-to-frame variation in target height exceeds the dynamic threshold. The l_1-tracker takes over going ten frames back, whose results are indicated in Fig. 5 (e)-(j). Eventually the CamShift tracker is able to recover when the occlusions in the intersection have passed, and the l_1-tracker is shut down. The reinitialized CamShift results are shown in Fig. 5 (k)-(l), from the last result by l_1 up the next occluding tree. This occlusion is detected from the difference in the distance associated with the left and right sides of the target exceeding 75% of the maximum depth difference from minimal to maximum disparity. This criterion was chosen experimentally for tolerance to noise within the depth map, but still easily picks up the difference between the depth associated with the road and the occluding tree.

The l_1-tracker is invoked once again to handle the occlusion as shown in Fig. 5 (m)-(n), and passes control back to CamShift, when the occlusion has passed, as seen in Fig. 5 (o), until the car goes out of frame. Altogether, the sequence is shown overlaid on the depth map in Fig. 4.

(a) Initial frame for all tracking sequences, Frame 01850, l_1



(b) Frame 01890, l_1

(c) Frame 01930, l_1

(d) Frame 01970, l_1

Fig. 3 Results for l_1 tracking only, every fortieth frame at 30fps. The series of occlusion eventually disrupt the template set to the point it completely loses the target 2 frames after (d), 123 frames in total

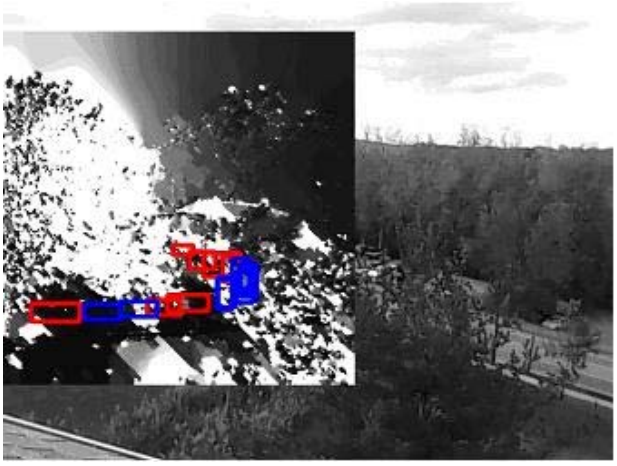


Fig. 4 Sequence of tracking results, every ten frames, overlaid on the depth map

VI. Conclusion

A hybrid tracking scheme has been demonstrated that is robust to both drift and occlusion and nominally computationally efficient. The principle algorithm combines the CamShift algorithm incorporating depth information using stereo vision with the robustness of l_1-minimization tracking in a particle filter framework. The novelty of this work is in combining the two core algorithms of tracking - the CamShift kernel-based and the l_1 state space-based method - and in formulating a control scheme for passing control from one to the other. The occlusion detector based upon the use of depth information could be implemented on other tracking algorithms as well, independent of any hybrid scheme. The main weakness of the algorithm is that some parameters need to be empirically derived to tune tracking performance.

VII. Future Work

The hybrid algorithm could be improved by adding additional criteria to interchange control, such as conducting recursive state estimation in a unified context between component algorithms, instead of initializing and shutting down the particle filter with l_1-tracking, as presently implemented. In addition, the target candidates from CamShift could be used as templates for similarity comparisons to the l_1-tracking results, for more robust detection of when an occlusion has actually passed. Both of these would improve the criteria for control transfer from l_1 to CamShift, and mitigate the likelihood of drift due to the coordinates of the occlusion, instead of the target, getting passed to CamShift. Measurement of the target size in physical units is noisy and drift could occur if the occlusion were of similar size as the target, as well as if similar appearance and trajectory.

Implementing the l_1-tracker portion in C++ OpenCV would have performance advantages in addition to the practical advantage that MATLAB core licenses would not be required for operation. It is expected that such contribution to the OpenCV community will pave the way for a plethora of hybrid trackers similar to the one developed, in the same way that the OpenCV implementation of CamShift paved the way for many combinations and variations of trackers. The method proposed here can easily adapt other combination tracking algorithms with l_1-minimization, for super hybridized tracking algorithms.

(a) Frame 01850, CamShift

(b) Frame 01860, CamShift

(c) Frame 01870, CamShift

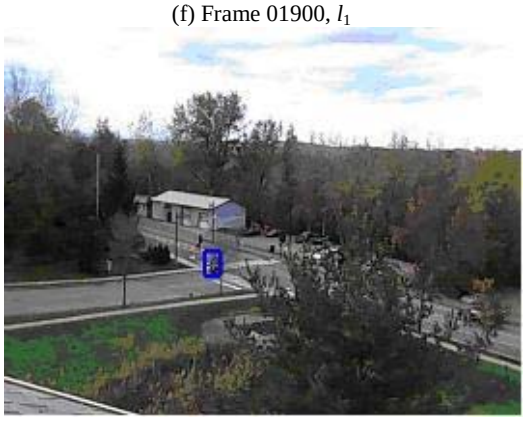
International Science Index, Computer and Information Engineering Vol:7, No:1, 2013 waset.org/Publication/1686

International Science Index, Computer and Information Engineering Vol:7, No:1, 2013 waset.org/Publication/1686

(d) Frame 01880, CamShift

(e) Frame 01890, l_1

(f) Frame 01900, l_1

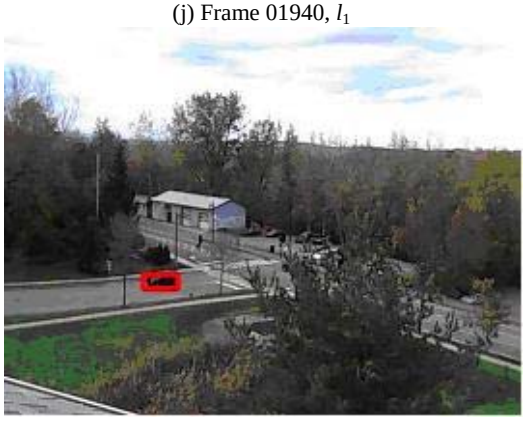


(g) Frame 01910, l_1

(h) Frame 01920, l_1

(i) Frame 01930, l_1

(j) Frame 01940, l_1



(k) Frame 01950, CamShift

International Science Index, Computer and Information Engineering Vol:7, No:1, 2013 waset.org/Publication/1686

(l) Frame 01960, CamShift

(m) Frame 01970, l_1

(n) Frame 01980, l_1

(o) Frame 01990, CamShift

Fig. 5 Hybrid tracking sequence results (boxed) for a car turning under occlusion, every ten frames at 30fps. Sequence ordered (a) to (o). Tracking results indicated in red (gray) boxes for CamShift or blue (black) boxes for l_1 as part of hybrid sequence

ACKNOWLEDGMENT

Supported by a VSGC VT-NASA EPSCoR Grant.

REFERENCES

[1] Bradski G., Kaehler A., "Learning OpenCV: Computer Vision with the OpenCV library", *O'Reilly Media, Inc*., 2008.
[2] Bruns E., Kurz D., Grundhiver A., and Bimber O., "Fast and robust CamShift tracking", *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshop*, June 2010, pp. 9-16.
[3] Wang Z., Yang X., Xu Y., Yu S., "CamShift Guided Particle Filter for Visual Tracking," *Journal Pattern Recognition Letters*, Vol. 30, Issue 4, March 2009, pp.407-13.
[4] Emami E., Fathy M., "Object Tracking Using Improved CAMShift Algorithm Combined with Motion Segmentation,"*Conference on Machine Vision and Image Processing* , 16-17 Nov. 2011, pp.1-4.
[5] Zou T., Tang X., Song B. , "Improved CamShift Tracking Algorithm Based on Silhouette Moving Detection," *2011 Third International Conference on Multimedia Information Networking and Security*, 4-6 Nov. 2011, pp.11-15.
[6] Li J., Zhang J., Zhou Z.; Guo W., Wang B., Zhao Q. , "Object tracking using improved CamShift with SURF method," *2011 International Workshop on Open-Source Software for Scientific Computation*, 12-14 Oct. 2011, pp.136-141.
[7] Kovacevic J., Juric-Kavelj S., Petrovic I., "An improved CamShift algorithm using stereo vision for object tracking," *MIPRO 2011 Proceedings of the 34th International Convention*, 23-27 May 2011, pp.707-710.
[8] Mei X. and Ling H., "Robust Visual Tracking and Vehicle Classification via Sparse Representation", *IEEE Trans. PAMI*, Nov. 2011, Vol.33, Isssue:11, pp.2259-2272.
[9] Deilamani, M.J. and Asli, R.N., "Moving object tracking based on mean shift algorithm and features fusion", *Artificial Intelligence and Signal Processing*, 2011, pp.48-53.
[10] Comaniciu D. and Meer P., "Mean Shift: A Robust Approach Towards Feature Space Analysis", *IEEE Trans. PAMI*, May 2002, Vol.24, No.5, pp.603-619.
[11] Wright J., Yang A.Y., Ganesh A., Sastry S.S. and Ma Y., "Robust Face Recognition via Sparse Representation", *IEEE Trans. PAMI*, Feb. 2009, Vol.31, Issue:2, pp.210-227.
[12] Candes E., Romberg J. and Tao T., "Stable signal recovery from incomplete and inaccurate measurements", *Comm. on Pure and Applied Math*," 59(8), 2006, pp.1207-1223.
[13] Donoho D., "Compressed Sensing", *IEEE Trans. Information Theory*, 52(4), 2006, pp.1289-1306.

Clark Van Dam received the B.S.E degree in electrical engineering from the University of Michigan, Ann Arbor, MI in 2005 and the M.S. degree in Physics from Central Michigan University in 2009. Currently he is a Ph.D. candidate in electrical engineering at the University of Vermont.

Gagan Mirchandani, Professor, heads the Signal Processing Group in the School of Engineering at The University of Vermont. URL: http://www.cems.uvm.edu/~gmirchan/