A Robust Human Detection and Tracking System Using a Human-Model-Based Camera Calibration

Zhong Zhang, Peter L. Venetianer, Alan J. Lipton

A Robust Human Detection and Tracking System Using a Human-Model-Based Camera Calibration

Zhong Zhang, Péter L. Venetianer, Alan J. Lipton
ObjectVideo, Inc.
{zzhang, pvenetianer, alipton}@ObjectVideo.com

Abstract

We present a robust, real-time human detection and tracking system that achieves very good results in a wide range of commercial applications, including counting people and measuring occupancy. The key to the system is a human model based camera calibration process. The system uses a simplified human model that allows the user to very quickly and easily configure the system. This simple initialization procedure is essential for commercial viability.

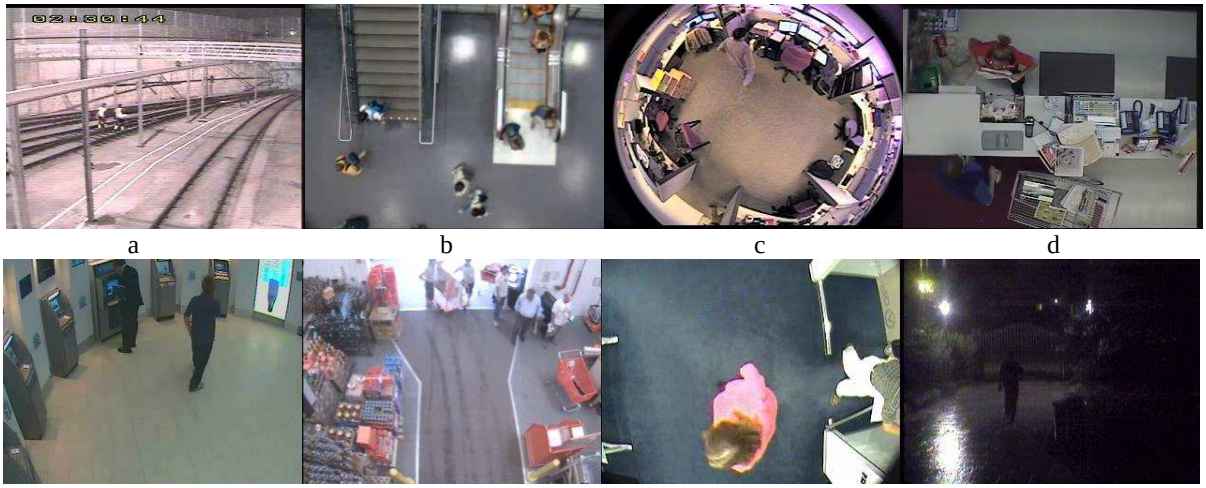
1 Background

More and more video surveillance cameras are installed every day. These cameras collect prodigious amounts of data which in most cases are analyzed only after the fact to aid forensic investigations. Intelligent video surveillance (IVS) systems [1][2] have the potential to overcome the data analysis problem, though their performance is still problematic in several application areas.

Two major application areas for IVS systems are security and business intelligence. The primary goal of security applications is to detect potential threats in real time, and alert responders to the danger in time to allow prevention. Forensic applications look for similar events, but after the fact. Security applications typically require very high detection and very low false alarm rates to be useful and commercially viable. Typical applications for IVS systems include detection of perimeter intrusion, suspicious object insertion (left bag), suspicious human behaviors (such as loitering), and illegal or suspicious vehicle parking events, just to name a few.

The purpose of business intelligence applications is the collection of statistical information over time. The output of such a system is typically some form of summary report. There is much less emphasis on individual events. While the underlying technology and the core events of interest are often similar between business intelligence and security applications, the requirements are somewhat different. Typically no real-time reporting is required, and the application is less sensitive to the accuracy of detecting individual events, as longs as the overall statistics are reliable. Typical applications include: detection of customer behaviors in retail environments (such as counting people in a store or determining shopping habits within a store); measurement of operational efficiencies such as using line queue length to determine staffing levels; and policy auditing such as making sure 2-person policies are adhered to.

Figure 1 shows some real world application scenarios. Figure 1a shows a security application detecting humans on the tracks while ignoring other legitimate targets such as trains. In Figure 1b the system counts people moving down from the escalator and the stairs while ignoring all other traffic. Figure 1c demonstrates monitoring occupancy in a commercial space, providing useful information for building management applications such as automatic heating, ventilation and air conditioning and lighting control applications [3]. The system of Figure 1d combines information from video analytics with point-of-sale transaction information in a retail environment to detect fraudulent or anomalous activities, such as returning an item without a customer being present [4]. Figure 1e shows loitering detection in front of an ATM. Figure 1f shows a queue length monitoring application reporting how many people are in line and for how long. Figure 1g illustrates access control “tailgating” detection where the video surveillance system can be integrated with a keycard reader to ensure that only one person enters for every card swipe. Figure 1h shows a residential security application, alerting when suspicious human activity is detected, while ignoring all non-human targets.



a b c d

e f g h

Figure 1: Some real world application scenarios

These applications may have quite different objectives and camera views may also look very different but they have one thing in common: they all require robust human target detection and tracking for reliable intelligent video analysis.

The demand for such intelligent video systems (IVS) is rising rapidly, but unfortunately, commercially viable solutions are often inadequate due to limitations with the existing video content analysis technologies. To provide a reliable, commercially viable computer vision based solution for a general user, the system must meet the following requirements:

1. Detect and track individual human targets even in crowded scenes;
2. Distinguish human targets from non-human targets;
3. Be robust to illumination changes and camera locations and types;
4. Operate in real-time with a simple, user friendly configuration.

Most "human detection" algorithms [5][6] use appearance-based approaches and usually require training in advance. These approaches usually have one or more of the following limitations:

- They are highly view dependant, and usually rely on more oblique camera views and specific human models to recognize humans. Thus they tend not perform well for overhead camera views.
- Some are single image-based. These approaches only provide an instantaneous count of the number of human targets within a single frame, and do not provide information on the motion behaviors of each target, thus do not solve most of the problems in Figure1.
- They do not handle target occlusion well. Human targets are often misdetected while fully or partially occluded by other targets or at the periphery of the video frame.
- Their training processes can be quite involved and significantly increase installation and maintenance costs.

In this paper, we introduce a robust human target detection and tracking approach that works in various real world environments and requires limited user interaction. Section 2 provides an overview of the whole system. Sections 3 and 4 give a detailed description of the technical approach. Experimental results as well as some real world application examples are given in Section 5. Finally, conclusions are presented in Section 6.

2 System Overview

A typical IVS system operates as follows [1][2]: a dynamic background model is continuously being built and updated from the incoming video frames. Pixels that are different from the background are marked as foreground. In addition, frame differencing can also be used to detect foreground pixels. These foreground pixels are spatially grouped into blobs. The blobs are tracked over time to form spatio-temporal targets. These targets are finally classified.

The system described in this paper largely follows this architecture. However, it differs from typical systems in the way it detects and tracks human targets. Instead of simple spatial grouping of foreground pixels into blobs, it uses calibration information and a human model to detect humans even in crowded scenarios.

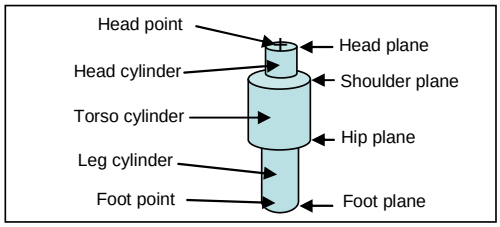


Figure 2: A general 3D human target model

The following sections describe in detail how this is achieved.

3 Calibration and Human Modeling

We propose a robust and efficient human detection and tracking system that relies on a simple calibration procedure as the only user input and has shown very good results in a wide range of field installations. In order to simplify computations the system makes the following 4 assumptions that were found to be true in many practical applications, thus imposing few limitations on the usability of the system:

1. All targets of interest are human;
2. All targets are on the same ground plane;
3. The camera is stationary, with no significant roll (i.e mounting without significant rotation around the optical axis), mounted higher than the monitored targets, and looking down on them;
4. The image center is the camera view center.

The first component of the proposed human detection and tracking system is a human model. This model consists of two parts: a simple 3D human target model representing a generic human shape, and a representation of the human model projected into the video image via camera calibration. This projection specifies how the humans are expected to look in the video. This section describes these components in detail.

3.1 Human target model

Our system relies on a simplified human target model, representing an average sized human. The model consists of 3 cylinders, bounded by four critical planes and 2 critical points, as illustrated in Figure 2. The physical size of the 3 cylinders is listed in Table 1.

The mapping of the 3D human model of Figure 2 to the 2D image is approximated by mapping the convex hull and certain pivot points, as illustrated by Figure 3. The pivot points include the foot point and some points selected from each of the four critical planes. The image based centroid of the convex hull is also computed.

Table 1: The default size of the cylinders of the 3D human target model

	Head	Torso	Leg
Height	27cm	70cm	79cm
Radius	9cm	21cm	12cm

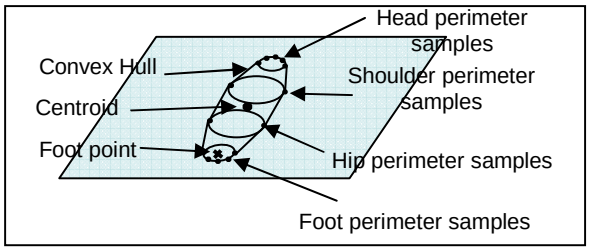


Figure 3: Human target projected onto the image plane

3.2 Human-based camera calibration model

The purpose of establishing the mapping between the world and the image coordinates is to be able to compute the transformation between the 3D human model of Figure 2 and the 2D image. As a result the system knows the expected average human size and shape at every image location. This is achieved by partially calibrating the camera.

There are several known methods to calibrate a camera [10][11], but most of them are prohibitively expensive and complex to perform, thus infeasible for a large scale commercial installation. [12] calibrates the camera based on data extracted from pedestrians, but it cannot handle overhead camera views and occlusion and shadows make the underlying automatic head and foot position detectors inaccurate. In the present work, we propose a fast method that requires only minimal user interaction, yet computes all the parameters required for mapping the 3D human model to the 2D image plane. Assumptions 3 and 4 of section 2 lead to the conclusion that only four calibration parameters are required: 2 external (camera height relative to the ground plane H_c, and tilt angle α_t relative to vertical) as illustrated in Figure 4 and 2 internal (camera focal length f_c and radial lens distortion K).

Given an ideal pinhole camera and the first 3 of the camera parameters (excluding radial distortion), a 3D world point (X, Y, H) can be mapped to an image point (x, y). An image point can be transformed back to a 3D world point if its height H is known. In our system H is typically assumed to be the height of a human target, and is normalized:

Now consider the fourth calibration parameter: lens distortion. In this application, a first order radial lens distortion model works well:

$$r' = r + K*r^2$$

where r and r' are the distance to the image center without and with lens distortion, respectively, and K is the lens distortion factor. An undistorted image point (x,y) is transformed to a distorted point (x',y') using

$$(x', y') = (xr'/r, yr'/y)$$

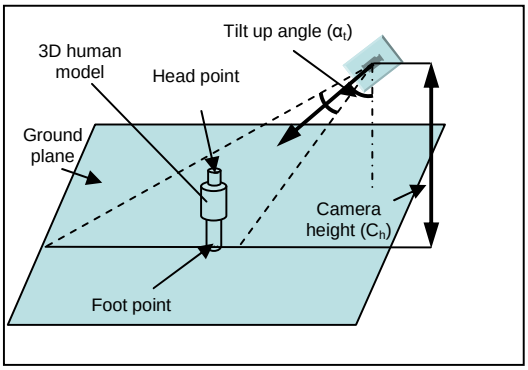


Figure 4: Parameters of the camera model

3.3 Camera calibration

The previous section describes the required camera calibration. So the question is: How is calibration information obtained? Given that installation must be simple, it is unrealistic to expect installers to accurately measure camera parameters such as height, tilt angle, focal length, and radial distortion. The method described in this section requires minimal user interaction, no training and is computationally simple.

Computing lens distortion during calibration requires lots of data and a complex, involved procedure. But during extensive testing we determined that the lens types used in practical applications can be divided into 5 groups, and an optimal K value can be selected for each group, as listed in Table 2. As the first step of configuring the system, the user is asked to select one of the 5 camera types. To extract the remaining three calibration parameters, the method needs 3 features on the image of a standing person at 3 or more positions, as illustrated on Figure 5. The three

Figure 5: Calibration images: 3 features (bounding box, head, foot) marked on each target used for calibration

Table 2: Lens distortion types with corresponding lens distortion factors assuming a 320x240 image

Lens type	Distortion parameter	Image observations
Pinhole	0	Some synthetic video or very high quality video with no image distortion observed or camera FOV < 30 degree
Normal lens	-0.00062	Camera FOV < 60 degree with not very noticeable image distortion
Normal wide angle	-0.001	60<FOV<120; with obvious image distortion
Fisheye	-0.0015	FOV>120 with severe image distortion
Black corner fisheye	-0.0019	FOV>170 with black corners on the image frame due to the severe lens distortion

target features used are the bounding box; the foot location and the head location. In practice, this data is collected from a human operator using a simple GUI, but it could be done by automatically detecting and tracking people moving alone in the scene.

The camera parameters are estimated using an iterative process. After the lens type and the sets of human calibration features are specified, the potential parameter range is estimated to speed up subsequent steps by reducing the search space. The range of the tilt angle α_t is bounded by the basic assumption that the camera is not looking up, limiting it to the (-90,90) degree range. This range may be further constrained by considering the relative position of the head and foot points: If the head and foot points are located close to the center of the bounding box, the camera is likely looking down corresponding to a small α_t. If the head and foot points are located close to the boundary of the bounding box, the tilt angle α_t may be relatively large. The range for the focal length f_c and the camera height H_c can both be bounded by the human sample size. The values are also closely related, with smaller focal length having a similar effect to increased height.

After the maximum parameter range has been determined, the three parameters are estimated using a coarse-to-fine brute-force search through the parameter space, taking a few seconds. This is acceptable because calibration is an off-line process and does not require a real-time implementation. First, the calibration method iterates through the parameter range using coarse steps, followed by finer ones. For each iteration i a normalized matching error metric is computed as the weighted average of the head/foot and the bounding box matching errors:

$$E(i) = w_{hf} * E_{hf}(i) + w_{bb} * E_{bb}(i)$$

The normalized head-foot matching error E_{hf} is computed as the average Euclidean distance between the head and foot points of the observed sample and the corresponding head and foot points of the projected 3D reference human model, normalized by the mean radius of the observed sample and the reference sample. The radius r of the observed sample is defined using the width w_b and height h_b of the bounding box as

$$r = \sqrt{w_b * h_b / \pi}$$

The bounding box matching error E_{bb} is determined as:

$$E_{bb} = 1 - (B_{obs} \cap B_{ref}) / (B_{obs} \cup B_{ref})$$

where B_{obs} and B_{ref} refer to the bounding box of the observed and the reference human samples, respectively.

The weights w_{hf} and w_{bb} are computed using the diagonal length of the observed bounding box D_{bb} and the image distance between the head point and foot point D_{hf}:

$$w_{hf} = 0.5 * (D_{hf} / D_{bb} + 0.5)$$

$$w_{bb} = 1 - w_{hf}$$

The iterations continue till the error is below a predefined threshold. The system provides visual verification of the accuracy of the calibration using a GUI superimposing calibrated human sizes on an image, as illustrated in Figure 6.

Figure 6: Visual verification of calibration accuracy: (a) user specified (blue) and corresponding computed (pink) features; (b) user specified calibration features (white) and computed features (dark blue) superimposed on a human target not used during calibration

3.4 The human model map

Based on the calibration information and the 3D human model, the system can compute what a human target should look like for every image pixel. This is precomputed off-line and stored in a human model map. The map contains, for every pixel, the convex model, the head and footprint locations in the image, the 3D footprint location, plus a human visibility ratio, which captures the percentage of a human that is visible at a given location. Figure 7 shows some human models as stored in the maps. The size of the people used during calibration determines the camera calibration parameters and the size of the stored human models. The system compensates for the typical variation in the actual size of the detected humans, though huge variability, e.g. mixing adults and kids can be problematic.

The human visibility ratio is used for two purposes. First, it helps eliminate false targets in image regions where there should be no targets, like the black area in Figure 7a. This not only reduces false alarms, but computational costs as well. Second, it provides the information on whether an observed image target represents a full human object or just part of a human object. This information is used for human target tracking described in Section 4.

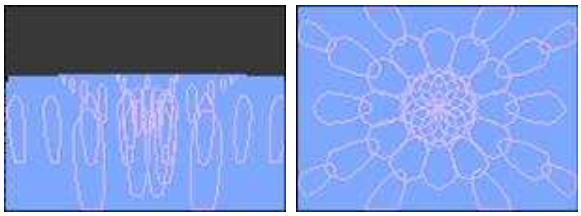


Figure 7: Human model maps: (a) oblique camera view with upright human models and a black portion where no humans are expected (above the horizon). (b) overhead camera view, where the expected size and shape of a human target can be quite different at different image locations.

4 Human target detection and tracking

In this section, we explain how the human calibration information described in Section 3 is integrated into the generic IVS framework of Section 2, and how it is used to overcome various challenges. The IVS system extracts various masks and foreground blobs for each frame. The flow chart of Figure 8 outlines how this input is used in our human target detection and tracking system.

First, a target prediction module is used to create the predicted human target model for each tracked target. A Kalman filter is used to estimate the 3D target footprint location. The corresponding 2D convex model (Figure 3) is computed based on the camera calibration.

Next, these predicted targets are matched with the foreground blobs detected in the current frame. A matching score M is used to measure how well the observed human target (the blob) matches the predicted human target model:

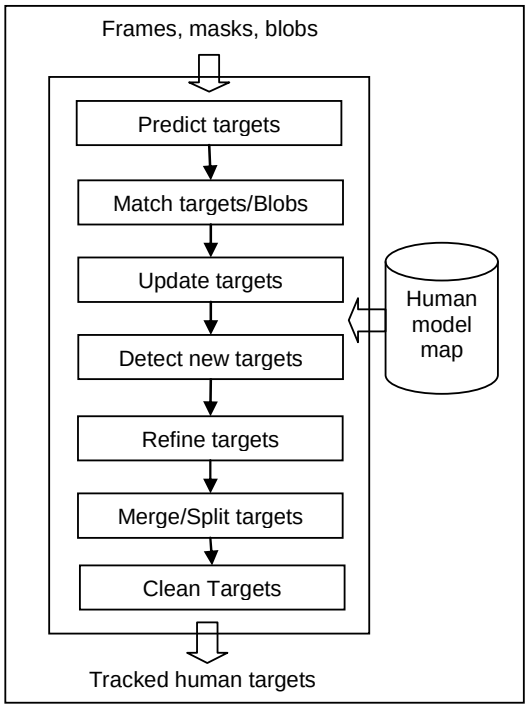


Figure 8: Block diagram of the human target detection and tracking module

$$M = R_{recall} + R_{precision} - R_{overlap} \quad (1)$$

where R_{recall} refers to the percentage of the blob area that is covered by the human target model, $R_{precision}$ is the percentage of the human model that is covered by the observed target and $R_{overlap}$ is the percentage of the blob area that is covered by more than one target model. In case of a one to one match, $R_{overlap}$ is 0. The association between the predicted human models and corresponding blobs is determined by maximizing the overall matching score for all the targets.

Next, human targets that have matching blobs are updated using these matching blobs as new observations. This updating combines the current observation and the prediction. An important problem is how to weigh these two against each other? The system uses two internal states, the target motion and the target stability states, defined below, to determine the weighting.

The target motion state reflects the moving status of a target and can be represented by a 3 state finite state automaton (FSA): *"Moving" state:* the centroid of the target is changing; *"Stopped" state:* the image centroid of the target is stable, but motion is detected within the target (internal motion); *"Stationary" state:* the target is completely motionless.

The motion state is determined based on the target trajectory (moving vs. not moving), and based on internal motion computed from frame differencing (stopped vs. stationary).

The target stability state reflects the confidence of the matching between the prediction and observation and can be represented by a 4 state FSA: *"Appearing" state:* the human visibility ratio or the observed human recall ratio is increasing. This represents that the target is entering the camera view or coming out from behind some occluding background object.; *"Disappearing" state:* the human visibility ratio or the observed human recall ratio is decreasing.. It indicates that the target is leaving the camera view or becoming occluded by a background object; *"Occlusion" state*: multiple targets match to the same blobs. This indicates that the target is occluding or occluded by other targets; *"Stable" state:* consistent matching between observation and prediction.

The tracker will assign higher weight to the observation in the "appearing" and "stable" states, and will emphasize the prediction more in the "disappearing" and "occlusion" states.

Blobs that didn't match any existing target are turned into new targets if they satisfy the following criteria, ensuring high confidence in the new targets:

Size and shape criterion: the blob must have a good matching score with the corresponding human model.

Visibility criterion: the corresponding human visibility ratio must be above a threshold. This ensures that a target is detected only when mostly visible, hence the system is more confident that it is not a non-human target.

Motion salience criterion: the human target must be moving into the camera view, not suddenly appearing. This helps avoid the detection of non-human targets.

Non-occlusion criterion: the new human target must not be in occlusion with other existing targets.

The goal of the "Refine Target" module is to reevaluate the observed blob and potentially change it to improve the matching score M between the observed blob and the corresponding human model. In (1), A low $R_{precision}$ means that the observed target is much smaller than the model, so the blob may need to be expanded using the surrounding less salient change or motion pixels. A low R_{recall} means that the observed target is much larger than the model, so some less salient pixels at the boundary of the blob might have to be filtered out. These pixels are typically caused by shadows and reflections.

The observed blobs can sometimes correspond to more than one human, or they may contain only part of a human. The goal of the "Merge/split targets" module

is to correct these mistakes by splitting up blobs that are too big and merging the ones that are too small.

A target with low R_{recall} and high $R_{precision}$ is candidate for splitting. A target will be split into two if both of the two new targets satisfy the new target criteria and the matching score between the blob and the targets improves significantly. For simplicity, the system never splits a target into more than two pieces in a single step. But big blobs will continue to be split into smaller pieces in subsequent frames.

Two targets with high $R_{overlap}$ value are candidates for merging. They are merged if the merge increases the matching score. To prevent multiple people in occlusion from merging together, both targets must be in moving motion state for a certain duration. This is based on the observation that the likelihood of a moving human target consistently be fully occluded by another moving human target is very small.

Finally, the "Clean Targets" module removes targets that disappeared from the camera view or those determined to be non-human. A disappeared target is one with no matching blob. The human visibility ratio is used to determine how quickly to remove the disappeared targets. If the human visibility ratio is low when last seen, and the target stability state is "disappearing", it indicates that the target is at the boundary of the camera view and leaving, thus it can be removed right away. If the human visibility ratio is high when last seen, it is more likely the target is blocked by some background object such as a tree. The system keeps predicting the target for some time to see if it reappears from the occlusion.

As mentioned earlier, the system assumes that all moving targets are humans, and there are no other moving targets like vehicles or animals. However, in some applications there are often other inanimate targets that move temporarily as a result of some human action. An example is a swiveling chair, which moves for a while after a person gets up from it. This type of non-human target might be mistakenly detected and tracked as a human target. The target motion state is used to differentiate between a chair with a person in it from a chair that is moving after the person left. The motion state of a human target rarely stays "stationary", even when not moving. It typically alternates between "stopped" and "stationary" as a result of frequent hand and/or head movements. In contrast, an empty chair or an abandoned baggage will quickly transition from the "stopped" into the "stationary" state and stay there afterwards.

5 Experimental results

Since we are building a commercial system, our goal was to develop an algorithm that will work robustly and run in real time. We implemented the above described algorithms both on a PC (in C++) and on a DSP (in C). A 320x240 video is processed on a 2.8GHz Pentium at 50fps, allowing 4 sensors to run simultaneously. On the Texas Instruments TMS320DM642/600 platform the system runs at 10fps

We tested system performance in a wide range of applications, both in the lab and in customer deployments. The two most frequently used applications were counting people entering/exiting an area using a virtual tripwire, and computing occupancy in an area of interest. In all these applications the system was installed using standard, often pre-existing cameras. Configuration consisted of only the calibration, as described in this paper, and setting up rules such as a tripwire for counting or an area of

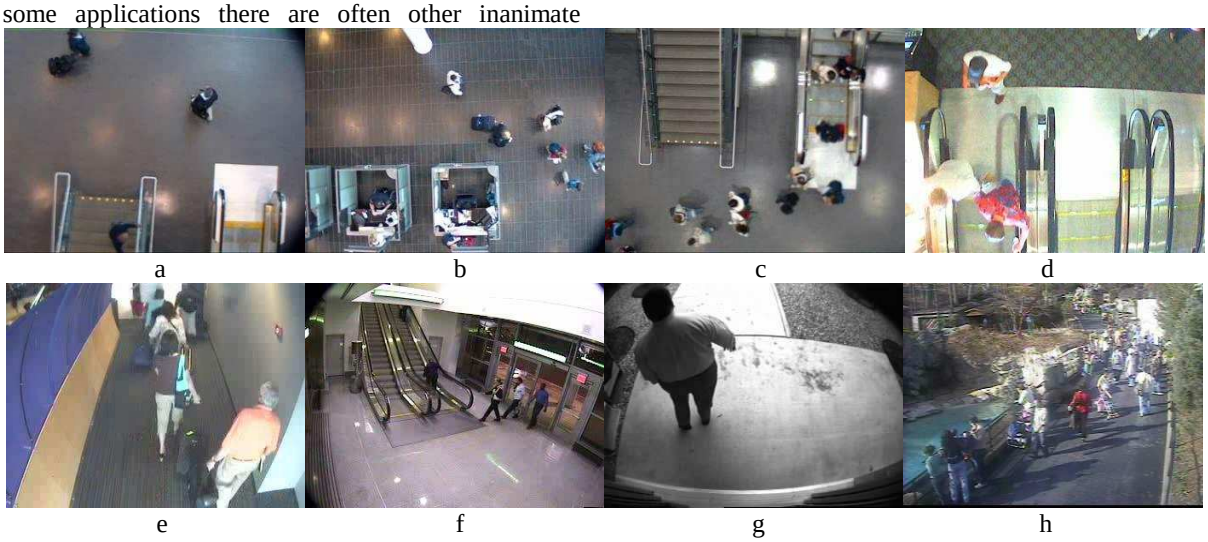


Figure 9: Examples of tripwire based counting scenarios

Table 3: People counting results from commercial deployments

Camera views	Ground truth count	System count	Accuracy (%)
A	16	17	93.8
B	102	98	96.1
C	516	500	96.9
D	45	41	91.1
E	54	41	75.9
F	306	297	97.1
G	564	597	94.1
H	9523	9031	94.8

interest for occupancy. While some published algorithms may outperform out results on certain clips, we couldn't find any method that demonstrated such consistent and high performance on such a large dataset.

Table 3 shows people counting results from a number of real world customer deployments. The camera views are illustrated in Figure 9. We groundtruthed 36 hours of video from these scenarios and measured counting accuracy by comparing the manual groundtruth count with that reported by the system. In most deployments the counting accuracy of the system was around 95%. The primary reason for undercounting was occlusions due to large crowds. This is even more pronounced with oblique views, like the one in Figure 9e. Overcounting was caused mostly by shadows and non-human targets mistakenly detected as humans.

To test the performance of occupancy counting, we again used data from a wide range of scenarios totaling about 10 hours, illustrated in Figure 10. Two metrics were used to evaluate performance. The first is the percentage of time the count is accurate (*%A*). The second is the mean difference between the ground truth and the computed occupancy count ($E(\Delta)$). Table 4 shows the results of the test, grouped by the number of people in the area. With less than 6 people in the view, the system tends to slightly overestimate, primarily due to shadows and non-human objects, such as moving chairs. For higher count, the system underestimates, mainly due to people occluding each other or people standing near the frame boundary.

Figure 10: Examples of occupancy counting scenarios: (a) conference room; (b) queue length

Table 4: Human occupancy counting results

Actual count	0	1-2	3-5	6-9	10+
%A (%)	96.6	97.4	99.1	95.2	85.2
E(Δ) (people)	+0.03	+0.04	+0.03	-0.33	-1.93

6 Conclusions

We presented the details of a human detection and tracking system. The system operates in real-time, and requires only a very simple calibration step for deployment, without the need for extended training period. We demonstrated very good performance in a wide range of application scenarios.

References

[1] R. Collins, A. Lipton, T. Kanade, "A System for Video Surveillance and Monitoring", Proc. American Nuclear Society (ANS) 8th Int'l Topical Meeting on Robotics and Remote Systems, 1999

[2] I. Pavlidis, V. Morellas, P, "Two examples of indoor and outdoor surveillance systems: motivation, design, and testing", Proc. 2nd European workshop on advanced video-based surveillance, 2001

[3] A.J. Lipton, J.I.W. Clark, B. Thompson, G. Myers, Z. Zhang, S. Titus, P.L. Venetianer, "The intelligent vision sensor: turning video into information", AVSS 2007

[4] P.L. Venetianer, Z. Zhang, A. Scanlon, Y. Hu, A.J. Lipton, "Video verification of point-of-sale transactions", AVSS 2007

[5] T. Zhao and R. Nevatia, "Tracking multiple humans in crowded environment", CVPR Vol. II: 406-413, 2004

[6] B. Leibe, E. Seemann, and B. Schiele, "Pedestrian detection in crowded scenes." CVPR 2005

[7] H. Elzein, S. Lakshmanan, and P. Watta, "A Motion and Shape-Based Pedestrian Detection Algorithm," Proc. IEEE Intelligent Vehicle Symp, 2003.

[8] P. Viola, M. Jones, and D. Snow, "Detecting Pedestrians Using Patterns of Motion and Appearance", ICCV, pp. 734-741, 2003.

[9] A. Shashua, Y. Gdalyaha, and G. Hayun, "Pedestrian Detection for Driving Assistance Systems: Single-Frame Classification and System Level Performance," Proc. IEEE Intelligent Vehicle Symp., 2004.

[10] R.Y. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the shelf TV cameras and lenses", IEEE Int. J. Robot. Automat.RA-3 (1987) 323–344

[11] G.-Q. Wei, S. De Ma, "Implicit and explicit camera calibration: Theory and experiments", IEEE Trans. PAMI. 16 (1994) 469–480

[12] F. Lv, T. Zhao and R. Nevatia, "Self-Calibration of a camera from video of a walking human", Proc. of the Int. Conf. on. Pattern Recognition, 2002.