Target tracking and surveillance by fusing Stereo and RFID information

Rana Hammad Raza*, George C. Stockman
Michigan State University | College of Engineering,
Engineering Building, 428 S. Shaw Lane, East Lansing MI 48824

ABSTRACT

Ensuring security in high risk areas such as an airport is an important but complex problem. Effectively tracking personnel, containers, and machines is a crucial task. Moreover, security and safety require understanding the interaction of persons and objects. Computer vision (CV) has been a classic tool; however, variable lighting, imaging, and random occlusions present difficulties for real-time surveillance, resulting in erroneous object detection and trajectories. Determining object ID via CV at any instance of time in a crowded area is computationally prohibitive, yet the trajectories of personnel and objects should be known in real time. Radio Frequency Identification (RFID) can be used to reliably identify target objects and can even locate targets at coarse spatial resolution, while CV provides fuzzy features for target ID at finer resolution. Our research demonstrates benefits obtained when most objects are "cooperative" by being RFID tagged. Fusion provides a method to simplify the correspondence problem in 3D space. A surveillance system can query for unique object ID as well as tag ID information, such as target height, texture, shape and color, which can greatly enhance scene analysis. We extend geometry-based tracking so that intermittent information on ID and location can be used in determining a set of trajectories of N targets over T time steps. We show that partial-target-information obtained through RFID can reduce computation time (by *99.9%* in some cases) and also increase the likelihood of producing correct trajectories. We conclude that real-time decision-making should be possible if the surveillance system can integrate information effectively between the sensor level and activity understanding level.

Keywords: Tracking, stereo, RFID, trajectories, frames, fusion

1. INTRODUCTION

Tracking of multiple objects in space is a fundamental problem with wide application and a rich literature. Our research interest is to monitor in real time the interactions of persons and objects using fusion of CV and RFID. It is significant for applications such as airport security, construction site safety, analysis of social or workplace interactions, analysis of games, patient safety and hospital asset management, old age home monitoring systems, and assisting persons with disability etc. Airport security in such crowded environments is of increasing importance that demands autonomous protection systems. Over the last decade, surveillance goals have changed from identifying individual suspects to social sorting of crowds based on perceived acute risk. This necessitates performing target[s] localization, tracking and activity analysis in real-time. Accomplishing these tasks using CV or RFID as stand-alone modalities is still an open research problem.

Many studies have focused on processing of video input to compute features used to extract separate moving objects[6,7,8,9]. A good survey of passive monocular methods is given by Veenman *et al*.[4]. Consistency of color, texture, shape, and motion can be used to track an object region across multiple video frames. Variable lighting, variable 2D projections of a 3D object, and occlusion of one object by another present difficulties. Applications that need to recognize what the objects are face additional uncertainty and complexity. For example, an autonomous vehicle needs to identify obstacles in its path using their image extent and their motion or apparent motion (see Otoom *et al*.[10]). McCoy *et al*.[13] investigated use of RFID for indoor airport security while dealing with issues such as target speed, tag orientation, effective read distance and target entry and exit etc.

Passive tracking algorithms using geometric constraints and naïve physics face exponential compute times, requiring use of heuristics. Even good tracking performance, however, cannot reliably provide object ID. In many workplaces it is possible for some agents and other objects to identify themselves – by special visual features (workers wearing yellow hats) or by radio frequency identification, or RFID (highway vehicle using E-ZPass to pay a toll), which can both increase reliability and decrease computational cost. Warehouse robotics systems from Litton Industries combine radio

*razarana@msu.edu; Phone 1 517 325-3260; ranahammadraza.com

Signal Processing, Sensor Fusion, and Target Recognition XXI, edited by Ivan Kadar,
Proc. of SPIE Vol. 8392, 83921J · © 2012 SPIE · CCC code: 0277-786X/12/$18 · doi: 10.1117/12.920526

communications between robots and controller, RFID on materials, and visual tracks on the floor to safely and efficiently control workflow[14].

An interesting example is the application of RFID and CV in a day-care environment (see Nakagawa *et al.*[2]). Parents can view their child's activity via the Internet. RFID tags are placed on the play objects and on the children so that readers within the play space can locate and identify them. Appropriate cameras can then be selected for good views of particular children and/or particular objects. Software alarms can be implemented for interaction between special pairs of objects and summarization of an entire day's activity can be done. There are other applications requiring similar functionality – elder care, studying how shoppers examine items for sale in a store, or how visitors examine art in a museum.

Over past decade, fusion of RFID and CV is also being used in indoor mobile and industrial robotics to support tasks such as autonomous recognition, localization and tracking. RFID alone has also been researched widely in this quarter. Passive stereo vision can locate detected objects in a 3D volume provided the image of the same object can be identified in two or more cameras[3]. An RFID reader can be used to ID an object observed in some 2D image, thus aiding stereo; or, a network of RFID readers can provide coarse 3D location without cameras. Thus RFID can help with object localization in multiple ways. RFID technology also enables smart objects to communicate information about themselves not available to optical sensors; for example object weight, container content, etc. A tagged rigid object can even help provide an optical observer with a network downloaded CAD model of itself to be used for pose computation by the observer. This was done by Hontani *et al.*[21], who also used visual tags on objects as a starting point in matching an observed image of the object to a projection of the 3D CAD model. Chae *et al.*[15] using fusion of RFID and CV proposed a global to fine localization algorithm for a mobile robot in an indoor environment. In a work space of *6.2 x 7.8 meters* they reported a mean localization error of *0.23 meters*. Another object localization scheme in a home environment using ceiling cameras is also presented[16]. Authors claim that use of RFID increased recognition accuracy by *34.5%* and reduced computation time by half. Lin *et al.*[17], reported fusion aided topological map-based navigation for mobile robot localization. Also the problem of dynamic obstacle recognition in mobile autonomous platforms is addressed by using fused information[18]. Limitations in mobile robots such as *dead reckoning* [in case of *robot kidnapping*] or *wheel slippage* are also addressable using fusion.

RFID provides indirect access to location information by using various localization schemes. Detail of these schemes are provided by Sanpechuda *et al.*[19] and Zhou *et al.*[20] and RFID localization accuracies are compared[19]. They range from *0.016 meters* and *0.026 meters* to *18 meters.* Within a controlled environment these techniques come with their drawbacks of high cost, large computation and setup time. Each RFID infrastructure and localization approach comes with its strengths and weaknesses. Combination of approaches can supplement the shortcomings of each other. Linearly scaling our stereo localization lab results present residual error of *0.7 meters* over a volume of *68 x 81 x 61 meters*. We have deduced that using fusion 3D location precision of about *0.5 meters* over *50 meters* should not be difficult to achieve.

In contrast to some other work, we assume in this paper that agents are *mostly known and cooperative*. We do not assume a controlled indoor environment, but do assume that a survey of the terrain exists including benchmark locations and that most agents have active RFID tags as well as distinctive clothing, which will often simplify object recognition. This partial control is needed since tracking outdoors presents difficulties with varying lighting, rain, smoke, dust, and noise, and occasional unexpected agents or objects. Moreover, real-time response is needed for safety.

In the sections that follow, we show that recognition of *some objects* during *some time intervals* can greatly speed up and make more reliable the organization of time frame information into the tracks of separate objects. Section 2 discusses our concepts and notation: note that these are presented in general terms and are not specific to a particular application. Section 3 describes the experiments we used to ascertain the benefit of some object ID information in tracking multiple objects. Section 4 discusses the experimental results in terms of our original objectives. Overall conclusions are given in Section 5, where we conclude that indeed partial object recognition from RFID or controlled CV can reduce both the computation time and uncertainty in multiple object tracking and thus enhance safety monitoring.

2. MODEL OF THE PROBLEM AND PROPOSED SOLUTION

In order to study the problem and to provide a solution that is independent of a specific application, we abstract the problem as follows.

Consider a database view with records $<x, y, z, t, L>$, each recording that an object with label (name) L is at location (x,y,z) at time t. This database is to be built from observations from RFID readers and/or a sensor network infrastructure together with networked stereo vision sensors. Higher level motion analysis will use this data and be triggered by daemons that monitor conditions in the data – e.g. nearness of objects of class *C1* and class *C2*. Higher level activity analysis is thus based on the real-time object track data. Without loss of generality, we continue our discussion using the site safety application.

The site-safety system *(S-3)* needs to identify and locate all significant objects in the workspace within a few frames k of real time observation. *S-3* may know $L = f(<x,y,z,t>)$ from sensor subsystems that use RFID or visual features. When such information is unavailable, the system can use “tracking” to determine $L = f\,(<x,y,z,t>)$ using prior records $\{<x, y, z, t\text{-}k>\}$, or perhaps even forward records $\{<x, y, z, t+k>\}$.

Tracking, which is the main concern of this paper, is a lower level of motion understanding that uses naïve physics to aggregate observations of N objects moving over T time frames. Heuristics from naïve physics enable aggregation of individual observations into a sequence or tracks, one for each moving object.

a. An object n must be at one and only one place at time t.
b. Location $<x,y,z>$ can accommodate at most one object at time t.
c. Object n is likely to have consistent form and visual features.
d. Observations of object n must be consistent with its identity, if known.
e. The motion of object n is likely to have smooth direction.
f. The motion of object n is likely to have smooth velocity [makes problem more complex].
g. Constraints e and f are likely to be violated only when object n is in close proximity to another object m.
h. Known objects are likely to move in a known terrain in predictable ways.
i. Some objects are known at some locations and time instants.
j. Objects do not enter or exit the workspace [our assumption]
k. Noise may add in input trajectory points during stereo calculation.

These constraints are an extension of those used by Sethi and Jain[1] and Veenman *et al*.[4] and, unfortunately, none are *hard constraints*. For example, it may be that constraint b) is violated as one object “consumes” another. Perhaps a driver enters a vehicle – which S-3 should prevent!

Our goal is to create a smart tracking algorithm based on the heuristics above, which will provide the means for safer activities and more efficient site management. Input to the tracker is a set of T vectors of information for each time frame $t=1,2,3,\ldots T$. Each of these “frame vectors” contains N tuples $<x,y,z, L>$, where label L may identify a known object $(L=1,2, \ldots,q \text{ or } N)$ or it may be unknown $(L=0)$. The purpose of the algorithm is to assign (discover) labels $L = 1,2,3 \ldots q$ *or N*, to each position tuple at each time t.

Our algorithm is motivated by the Sethi-Jain[1] and Veenman *et al.*[4] algorithms. It can work in either 3D or 2D and includes more information on some objects at some time instants [as is available from the RFID or vision sensors]. If a 2D algorithm is used, the constraint that two objects cannot be in the same location at the same time should be relaxed since it may just be that one object is occluding the other at some instant. The general algorithm will have different specializations depending on the application and how much sensor information and object constraints are available. For example, in the S-3 system, our cameras are calibrated to a surveyed 3D terrain, so if an image object is known, then an approximate 3D object location can be computed using the image from a single calibrated camera (we can just intersect a camera ray, or cone, with a bounding sphere in the 3D space). For the work reported here, we extract away such application detail and replace it by probabilities of knowing object ID over various periods of time. The algorithm described in Section 3.5 works along a forward path and does not have a greedy exchange loop as used by Sethi-Jain.

3. EXPERIMENTS

To study the value of fused sensor information in tracking multiple objects, we have generated many sets of ground truth data. The first type of data is extracted from a lab bench using stereo vision. The second type is generated artificially using mathematical curves. The tracking algorithm is then applied to the observations to segment them into separate object trajectories; the automatically derived trajectories are then compared to the ground truth trajectories to assess performance. Since it is impossible for us to gather the number of cases needed using real data, RFID is simulated -- in extracting trajectories, object location and ID are sometimes randomly provided to the tracking algorithm for some of the observations.

3.1 Generating real trajectories

To generate "real trajectories", we use 3D stereo. A colored sphere on a stick is moved by hand along a trajectory within a wireframe cube. (The cube is used for calibrating the cameras). The trajectory of the sphere yields T records $<x,y,z,t,L>$ for object track L at times $1,2, \ldots, T$. The experimenter then repeats using the stereo system to generate more trajectories until there are N of them, one for each object moving in the workspace: $L = 1,2,3 \ldots N$. Each of these N sequences is an "object track". If we have N object tracks, then there are 2^N subsets of these to choose for study. We have generated multiple tracks by varying the path and velocity through the workspace and also taking care to create some near collisions. Figure 1 shows a set of a few ground truth trajectories generated using our stereo rig.

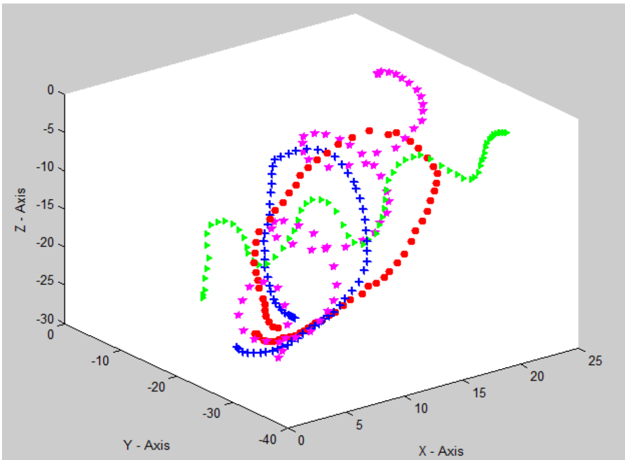


Figure 1. Example of ground truth trajectories

3.2 Generating mathematical trajectories

We created a data set generator that can randomly create smooth object tracks with various speeds and densities without collision. We generate N smooth paths for T time frames each in 3D space using a single helix, which is randomly spread out for a selected number of time frames using pseudorandom values as shown in Figure 2. The circular helix of radius a and pitch $2\pi b$ in 3D space is parameterized with Cartesian coordinates as follows:

$$x(t) = a\cos(t)$$

$$y(t) = a\sin(t)$$

$$z(t) = bt$$

To meet the constraints in Section 2, the generated data has the following parameters by default:

a. Object tracks $N=10$.
b. Time frames $T=11$.
c. Smooth velocity vectors.
d. Unique trajectory directions.
e. No chance of collision.
f. Randomly spread out trajectories in a 3D space of $1\ m^3$.

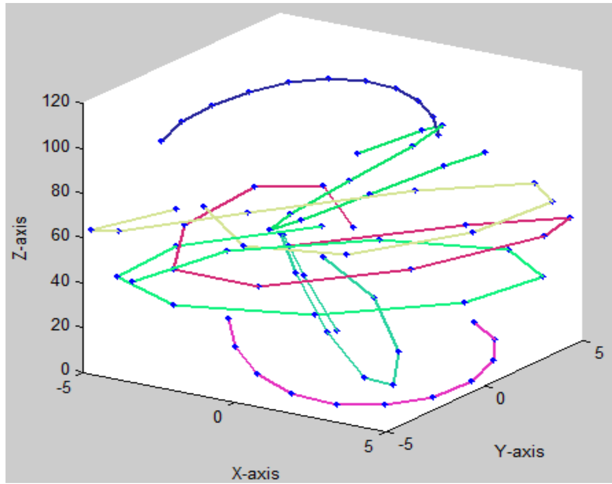
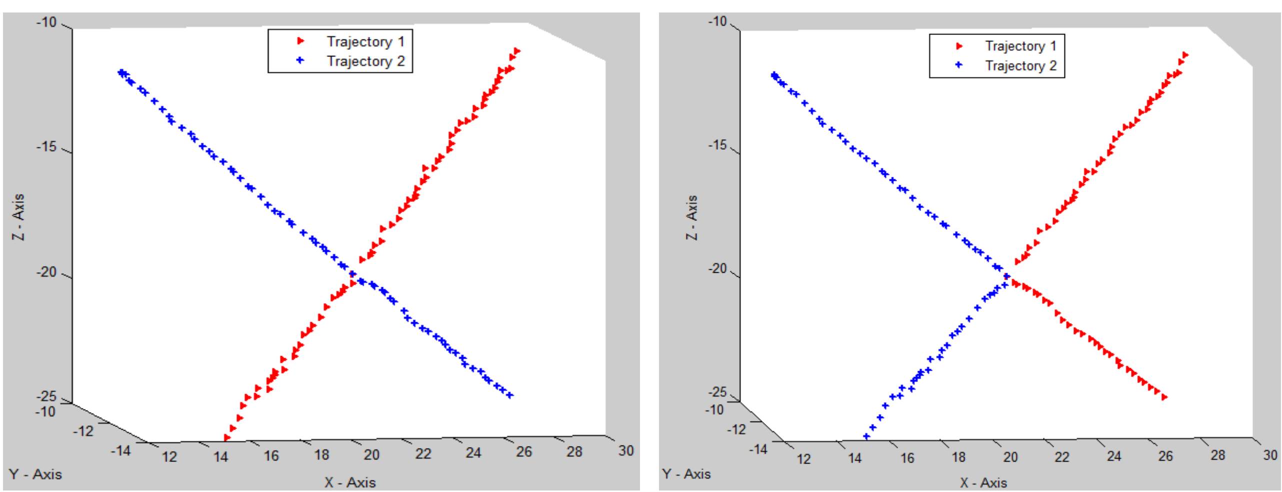


Figure 2. Generated trajectories with T=11 and N=7

3.3 Special cases

Special cases are revealing of algorithm behavior. Consider the case where two persons walk toward each other, exchange brief cases, and then backtrack to their original positions. Due to smoothness constraints, the geometric data will produce incorrect tracks with the persons continuing with their briefcases to different final positions. However, reliable location and ID of either person using either RFID or CV enables the correct interpretations to be extracted. This case is simulated by generating two ground truth trajectories *(N=2* for *T = 70* time frames) using the stereo rig. The point of intersection occurs at frame *t = 43*. The mean velocity of both the object tracks is kept the same. Using the smoothness criteria alone with no labeling information produced wrong trajectories as shown in Figure 3(a). However, once ID labels with location information are provided near the intersection, the tracking algorithm interprets correct object tracks as shown in Figure 3(b). To avoid ambiguities in the vicinity of collision points, some localization and object ID information is necessary outside the area of collision. Additional confounding factors will be considered in future work.



(a) (b)

Figure 3. (a) Wrong interpretation of trajectories with CV alone (b) Correct interpretation of trajectories with CV & RFID fusion

3.4 No object information versus some object information

Other crucial cases where sensors provide some location and ID information demonstrate how such information significantly reduces computation time. We tested tracking performance of the algorithm by generating ground truth trajectories in various configurations with the same mean velocities. One case amongst them had $N=3$ and $T=100$ time frames. The object labels were provided randomly at *15* time frames. This reduced execution time by *25%*. Results of many such experiments varying the amount of partial location and ID information are discussed in the next section.

3.5 Algorithm description

The algorithm takes as input N observations of 3D points over T time instants, thus NT tuples total grouped into T time frames. It extracts a smoothest set of paths through these points, observed in frames *1* to T; all tuples now grouped into N tracks. The object ID and location provide labeling information with the 3D points when available *i.e* $L = 1,2,3,\ldots\ldots,N$. The number of such "tagged points" must be less than or equal to T for any of the objects n. For object track n, the trajectory consists of 3D points at each time frame $t=1,2,3,\ \ldots\ldots\ldots,T$. The trajectory with label L is represented as:

$$C_L = [P_{n1}, P_{n2}, \ldots\ldots\ldots, P_{nT}]; \quad P_{n,t} = < x, y, z, t, L >$$

As in Sethi-Jain, the path difference between two consecutive 3D points is defined as:

$$D_{n,t} = P_{n,i} - P_{n,j}\ ; i \neq j \in t$$

Smoothness at a current point $P_{n,t}$ is calculated using the previous point $P_{n,t-1}$ and future point $P_{n,t+1}$. $D_{n,t-1}$ is the path difference between the current and previous point and $D_{n,t+1}$ is the path difference between current and future point. Smoothness value $S_{n,t}$ of a 3D point is then defined as follows:

$$S_{n,t} = w\left(\frac{D_{n,t-1}.D_{n,t+1}}{|D_{n,t-1}||D_{n,t+1}|}\right) + (1-w)\left(\frac{2\sqrt{D_{n,t-1}.D_{n,t+1}}}{|D_{n,t-1}||D_{n,t+1}|}\right)$$

To yield $0 < S_{n,t} \leq 1$ a weight factor w is used such that $0 < w \leq 1$. The initial points of N object tracks are assigned arbitrarily. The total sum of smoothness over T time frames for an object track n with assigned label L is then given as:

$$S^L_{Total} = \sum_{t=2}^{T-1} S_{n,t}$$

For efficient implementation, the algorithm uses a *block set* concept. A real time algorithm must make decisions within, say, a 5^{th} of a second, or *6* video frames. This limits the amount of look ahead that can be used. A *block set* denoted by B is defined as a group of N-tuples $<x,y,z,L>$ frame vectors for a fixed length of time frames m. The size of B is then $N \times m$. (In some simulations, we assume that the object ID and location are unknown [or known] for the entire group of m frames. From RFID properties, it is reasonable to assume that object IDs and their respective locations persist or are absent for multiple frames.)

Step by step implementation of the algorithm is as follows:

a. Input N observations of 3D points over T time instants.
b. For all N object tracks, assign labels $n = 1,2,3,\ldots\ldots,N$ arbitrarily to frame vector at $t=1$.
c. Using nearest neighbor assignments generate C_N trajectories for $t=2,3,\ldots\ldots,T$ time frames. Point $P_{n,t}$ in frame vector is assigned once to trajectory n in one time instance and cannot be reassigned elsewhere.
d. Consider m time frames at a time and loop over T time frames with increment of m-*1*, where, $m = 3,4$, *or 6*. This will form k time frame blocks. Each time frame block with m time frames and N assigned trajectories can now be represented as B_k i.e frame vector block set.
e. Compute all possible $|U|$ combinations of the elements of block set B_k. $|U|$ is $r \times m$. r is the product of the number of elements of N trajectories in block set B_k. Label L may identify a known object $(L=1,2,\ \ldots,q \text{ or } N)$ or it may be unknown $(L=0)$. Availability of partial label information will reduce number of possible combinations.
f. Calculate smoothness at every instance in m time frames for r combinations.
g. Calculate total smoothness S^r_{total} over m time frames for each combination of r.
h. Sort end total smoothness for each combination in descending order.

i. While indexing, choose highest total smoothness of N pairs with combinations having different elements in each frame vector in an instance.

j. Exchange points and assign c_N^k smooth trajectories in B_k as a subset of final smoothed trajectories C_N.

k. Return to step d and increment k. End loop until last positive integer value of k.

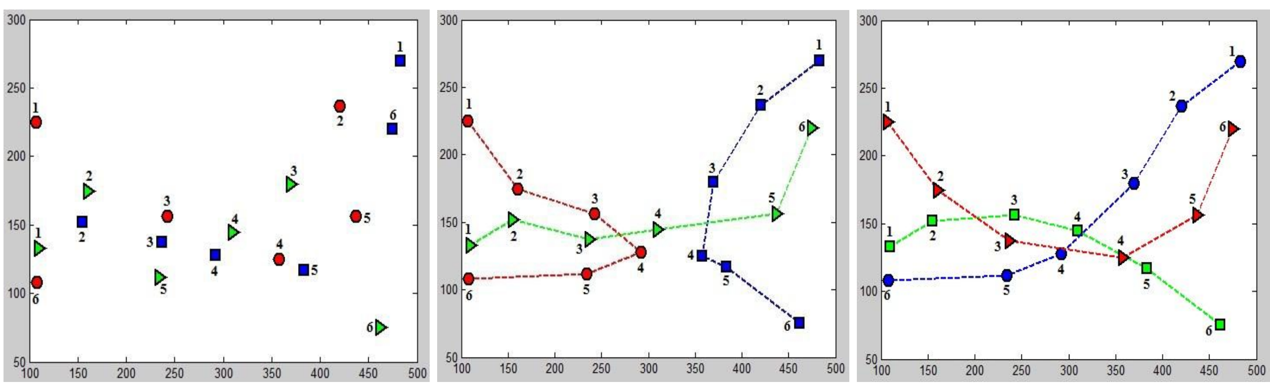
l. Correlate similar end points of c_N^k smoothed trajectories in B_k with similar initial points of c_N^{k+1} smoothed trajectories in B_{k+1}. Based on this similarity measure, rearrange the order/label of c_N^{k+1} smoothed trajectories.

m. Combine similar label subset trajectories from all frame blocks and generate final smoothed trajectories C_N.

4. DISCUSSION OF RESULTS

The tracking accuracy of the algorithm is assessed by a "track error" criteria. *Track error is defined as the fraction of wrong trajectory point assignments.* Objects do not enter or exit; therefore a wrong assignment between points $P_{j,t}$ and $P_{k,t}$ (where $j \neq k$) are considered as one error. The final track error is then averaged over the number of simulation runs. Alternatively, track error can be more fairly defined in terms of point sensing tolerance; an object label assigned to a sensed point is considered correct if the sensed point is within measurement tolerance of the ground truth sensed point.

To show the performance of the algorithm we demonstrate step by step results of an example. The input data is displayed in Figure 4(a) and consists of a sequence of *6* time frames with *3* trajectories having 3D data at each point. Each point of the trajectories is symbolized by ■, ● or ►. For better visualization the z dimension of all the input data is fixed. Figure 4(b) shows the trajectory assignments after nearest neighbor linking. In the next step the exchange candidates are then decided using total smoothness. For this example a frame vector block set B_k of length $m=6$ is used. Figure 4(c) shows the smoothed trajectories with ■ as label *1*, ● as label *2* and ► as label *3*. The algorithm took only *0.102 seconds* with *zero* track error.



(a) (b) (c)

Figure 4. (a) Input data (b) Nearest neighbor assignment (c) Smoothed trajectories

Table 1 shows the behavior of the algorithm in terms of track error performance. The simulations were done *using MATLAB® 2009* on *Core i5 M580 2.67 GHz* platform. The experiments were conducted while choosing randomly three real time 3D trajectories acquired from the stereo system. The simulations were run *20* times with different frame vector block set length m. Different values of time frames T were used. The outputs are then averaged to generate the results. The results were also compared to the ground truth. It is clear from Table 1 that varying m and density of points over time affects the track error performance. Since the stereo system readings generate an error of up to *10mm*, this error tolerance can be used in comparison to ground truth in determining track error. The last column of Table 1 shows the results with error tolerance applied while using $m=6$. Increasing m decreases the error, however, the number of possible combinations also increases so it affects computation time. These combinations can be reduced if we have partial knowledge of the trajectory points. To elaborate, Table 2 shows the possible combinations space with computation time at points of interest. The experiment was conducted using *N=5, 6* and *10* object tracks and *T=60* time frames with different probabilities of partial ID and location availability. The algorithm is run with *m=4*. Partial knowledge is assumed to be randomly available for *0.07 sec* per frame set over entire length of object tracks. This is justified in real time with *m=4* i.e. RFID feed once present stays for at least *0.07sec (4/60)*. Therefore, while tracking *10* objects the combinations space is decreased effectively up to *99.9%* with the presence of location and ID feed. The computation time is also reduced up to *99.9%* and therefore allows the algorithm to be executable in real time. It is also seen that the effect of partial information availability plays a vital role as the number of object tracks *N* increases. Also object location and ID info increase the accuracy of calculated trajectories.

Table 1. Track error performance with different points density and block length *m*.

	m=3	***m=4***	***m=5***	***m=6***	***m=6* *w/ error tolerance***
T=10+	*0.121*	*0.095*	*0.043*	*0.027*	*0.0*
T=20+	*0.196*	*0.096*	*0.054*	*0.042*	*0.004*
T=30+	*0.239*	*0.103*	*0.067*	*0.058*	*0.021*
T=40+	*0.251*	*0.134*	*0.083*	*0.066*	*0.023*
T=50+	*0.262*	*0.159*	*0.091*	*0.071*	*0.036*
T=60+	*0.266*	*0.167*	*0.098*	*0.075*	*0.038*

Table 2. Possible combination space with different *N* and various probabilities of object info availability.

Probability that observations have location and ID over T time frames	***Number of object tracks N with partial info***		
	5	*6*	*10*
0%	*9375 *3.9*	*19440*	*150000 *25.519*
26.7%	*2965*	*6248*	*46185*
53.3%	*663*	*1240*	*7832*
80%	*32*	*80*	*582 *0.73*

**computation time in seconds*

5. CONCLUSIONS

We provide a method to address the trajectory points correspondence problem in 3D space when partial object ID and location information is available. Our primary purpose was to analyze how partial object track information, which could be provided by RFID and network technologies, would make tracking more accurate and efficient. Our parameterization focused on airport security. The tracking algorithm was originally motivated by Sethi-Jain, though its working domain is in 3D and it does not use a greedy exchange step. Tracking algorithms sometimes lose correct object tracks at ambiguous intersecting points of trajectories. Providing partial information, such as object ID and location by RFID in such gray areas, helps the system to interpret correct trajectories. With such supplemental detail, algorithm exponential growth in computation with increasing *N* is also greatly decreased. While dealing with *10* object tracks for *60* time frames and having localization and ID randomly available *80%* of the time across all the input object tracks reduces the search size by about *99.9%*. This in turn reduces the execution time from *25 seconds* to only *0.73 seconds*. This reduction in time allows real-time implementation of the algorithm and improves on previously reported methods.

The algorithm currently doesn't allow object entry and exit and variable velocity at this time. These along with target to background and target to target occlusions are factors that we will incorporate in future work. Moreover, we need to improve the algorithm so that it can track *N=50* or more objects in real time.

The results we have shown apply to tracking in either 3D space or in a 2D image of that space. Using stereo in order to compute the 3D coordinates of objects implies that the correspondence problem has been solved, whereas other studies have used monocular tracking in order to assign these correspondences. Motion is just one feature that can be used in point matching. When unique features are available, they aid in stereo matching as well as in tracking over time. Application information also helps – we know the height of persons, the terrain on which they walk, their approximate speed, and occasionally their IDs from RFID. Sensor network technology is improving at a fast pace and we may even be able to use instantaneous object acceleration, temperatures, electrical field sensing, etc. Such information can be used at both the level of local object interaction and the level of overall site activity. Real-time decision-making should be possible if the overall system can integrate information effectively between the sensor level and activity understanding level. We have limited this study to examine the effect of partial object information on the performance of tracking. Integration into a practical system is left for future work.

REFERENCES

[1] Sethi, I. K. and Jain, R., "Finding trajectories of feature points in a monocular image sequence," IEEE Transactions on Pattern Analysis and Machine Intelligence, 9(1), 56-73 (1987).

[2] Nakagawa, S., Soh, K., Mine, S. and Saito, H., "Image systems using RFID tag positioning information," NTT Technical Review Journal, 1(7), 79-83 (2003).

[3] Shapiro, L. and Stockman, G., [Computer Vision], Prentice-Hall, Upper Saddle River, NJ, (2001).

[4] Veenman, C., Reinders, M. and Backer, E., "Motion tracking as a constrained optimization problem," Pattern Recognition, 36(9), 2049-2067 (2003).

[5] Verestoy, J. and Chetverikov, D., "Experimental comparative evaluation of feature point tracking algorithms," Proc. Workshop Evaluation and Validation of Computer Vision Algorithms, 183-194 (2000).

[6] Zheng, Q. and Chellappa, R., "Automatic feature point extraction and tracking in image sequences for arbitrary camera motion," International Journal of Computer Vision, 15(1-2), 31-76 (1995).

[7] Meier, T. and Ngan, K., "Automatic segmentation of moving objects for video object plane generation," IEEE Transactions on Circuits and Systems for Video Technology, 8(5), 525-538 (1998).

[8] Cucchiara, R., Grana, C., Piccardi, M. and Prati, A., "Detecting moving objects, ghosts, and shadows in video streams," IEEE Transactions on Pattern Analysis and Machine Intelligence, 25(10), 1337-1342 (2003).

[9] Zhou, Q. and Aggarwal, J., "Tracking and classifying moving objects from videos," Proc. IEEE International Workshop Performance Evaluation of Tracking and Surveillance, 52-59 (2001).

[10] Otoom, A., Gunes, H. and Piccardi, M., "Feature extraction techniques for abandoned object classification in video surveillance," 15th IEEE International Conference on Image Processing, ICIP 08, 1368-1371 (2008).

[11] Longbin, C., Feris, R., Yun, Z., Brown, L. and Hampapur, A, "Integrated system for moving object classification in surveillance videos," IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance, AVSS 08, 52-59 (2008).

[12] Wang, Y., Fang, S., Cao, Y. and Sun, H., "Image-based exploration obstacle avoidance for mobile robot," Control and Decision Conference, CCDC 09, 3019-3023 (2009).

[13] McCoy, T., Bullock, R.J. and Brennan, P.V., "RFID for airport security and efficiency," IEEE Seminar on Signal Processing Solutions for Homeland Security, 9-15, (2005).

[14] Egemin Automation, Holland, Michigan, USA; http://www.egeminusa.com/pages/agvs.hybrid.html

[15] Chae, H. and Han, K., "Combination of RFID and Vision for mobile robot localization," Proceedings of the International Conference on Intelligent Sensors, Sensor Networks and Information Processing Conference, 75-80 (2005).

[16] Kamol, P., Nikolaidis, S., Ueda, R. and Arai, T., "RFID Based object localization system using ceiling cameras with particle filter," Proceedings of the 2nd International Symposium on Smart Home, 37-42 (2007).

[17] Lin, W., Jia, S., Yang, F. and Takase, K., "Topological navigation of mobile robot using ID tag and web camera," Proceedings of International Conference on Intelligent Mechatronics and Automation, 644-649 (2004).

[18] Jia, S., Sheng, J. and Takase, K., "Obstacle recognition for a service mobile robot based on RFID with multi antenna and stereo vision," Proceedings of the IEEE International Conference on Information and Automation, 125-130 (2008).

[19] Sanpechuda, T. and Kovavisaruch, L., "A review of RFID localization: Applications and techniques," 5th International Conference on Electrical Engineering, Electronics, Computer, Telecommunications and Information Technology, ECTI-CON 08, 769-772 (2008).

[20] Zhou, J. and Shi, J., "RFID localization algorithms and applications - A review," Journal of International Manufacturing, 20(6), 695-707 (2008).

[21] Hontani, H., Nakagawa, M., Kugimiya, T., Baba, K. and Sato, M., "A visual tracking system using an RFID-tag," In Proceedings of SICE Annual Conference, 2720–2723 (2004).