International Journal of Pattern Recognition
and Artificial Intelligence
Vol. 28, No. 1 (2014) 1455001 (28 pages)
© World Scientific Publishing Company
DOI: 10.1142/S0218001414550015

World Scientific
www.worldscientific.com

ONLINE MULTIPLE TARGETS DETECTION AND TRACKING FROM MOBILE ROBOT IN CLUTTERED INDOOR ENVIRONMENTS WITH DEPTH CAMERA

YU ZHOU[*,§], YINFEI YANG[†,¶], MENG YI[‡,‖], XIANG BAI[*,**],
WENYU LIU[*,††] and LONGIN JAN LATECKI[‡,‡‡]

[*]*Department of Electronics and Information Engineering*
Huazhong University of Science and Technology, Wuhan 430074, P. R. China

[†]*Amazon, Seattle, WA 98105, USA*

[‡]*Department of Computer and Information Sciences*
Temple University, PA 19122, USA
[§]*yuzhou@hust.edu.cn*
[¶]*yangyin7@gmail.com*
[‖]*mengyi@temple.edu*
[**]*xbai@hust.edu.cn*
[††]*liuwy@hust.edu.cn*
[‡‡]*latecki@temple.edu*

Received 3 April 2013
Accepted 30 December 2013
Published 19 March 2014

Indoor environment is a common scene in our everyday life, and detecting and tracking multiple targets in this environment is a key component for many applications. However, this task still remains challenging due to limited space, intrinsic target appearance variation, e.g. full or partial occlusion, large pose deformation, and scale change. In the proposed approach, we give a novel framework for detection and tracking in indoor environments, and extend it to robot navigation. One of the key components of our approach is a virtual top view created from an RGB-D camera, which is named ground plane projection (GPP). The key advantage of using GPP is the fact that the intrinsic target appearance variation and extrinsic noise is far less likely to appear in GPP than in a regular side-view image. Moreover, it is a very simple task to determine free space in GPP without any appearance learning even from a moving camera. Hence GPP is very different from the top-view image obtained from a ceiling mounted camera. We perform both object detection and tracking in GPP. Two kinds of GPP images are utilized: gray GPP, which represents the maximal height of 3D points projecting to each pixel, and binary GPP, which is obtained by thresholding the gray GPP. For detection, a simple connected component labeling is used to detect footprints of targets in binary GPP. For tracking, a novel Pixel Level Association (PLA) strategy is proposed to link the same target in consecutive frames in gray GPP. It utilizes optical flow in gray GPP, which to our best knowledge has never been done before. Then we "back project" the detected and tracked objects in GPP to original, side-view (RGB) images. Hence we are able to detect and track objects in the side-view (RGB) images. Our system is able to robustly detect and track multiple moving targets in real time.

[**] Corresponding author.

The detection process does not rely on any target model, which means we do not need any training process. Moreover, tracking does not require any manual initialization, since all entering objects are robustly detected. We also extend the novel framework to robot navigation by tracking. As our experimental results demonstrate, our approach can achieve near prefect detection and tracking results. The performance gain in comparison to state-of-the-art trackers is most significant in the presence of occlusion and background clutter.

Keywords: Online tracking; robot navigation; depth camera.

1. Introduction

Online detection and tracking multiple targets in cluttered indoor environments are challenging tasks due to high intrinsic variation in appearance, shape, scale, pose, viewpoint, and extrinsic noise like illumination variance. In this paper, we present a unified framework that can be used for online category free object detection, tracking, and extend our framework to online, autonomous mobile robot navigation. Figure 1 gives the processing flow of the proposed approach. Figure 1(a) shows our mobile robot Pekee II[26] and its Kinect RGB-D sensor[27] mounted at 2 m height. Given a depth image (shown in Fig. 1(b)), we recover the 3D point cloud and detect the ground plane, which usually represents the floor. Then we compute a top-view of the scene, called GPP. To obtain GPP, we project all the 3D points to the ground plane, and remove the ground plane points themselves. Two kinds of GPP images are utilized in this paper: gray GPP (shown in Fig. 1(e)), which represents the maximal height of 3D points projecting to each pixel, and binary GPP (shown in Fig. 1(d)), which is obtained by thresholding the gray GPP. Detecting moving targets in GPP is a significantly simpler task than in the original RGB or depth images. The detected moving targets are back projected to the original RGB image (shown in Fig. 1(g)). Based on the detection results, a novel pixel level association (PLA) algorithm is proposed to estimate the motion of multiple targets, Fig. 1(h). Since we are able to not only detect moving objects but also static objects in GPP, our framework easily extends to online robot navigation by tracking. The path planning is performed completely in GPP, Fig. 1(i).

Compared with outdoor environment,[45] e.g. Fig. 2(a), the typical indoor environment, e.g. Fig. 2(b), is often more cluttered in side-view (RGB) images, the objects often occlude each other by the law of projective geometry, but when viewed from above, i.e. in GPP, they are clearly separated. Therefore, detecting and tracking targets in GPP is significantly easier than in the standard side-view images. In addition, benefit from GPP, the potential position and scale of the target can be predicted by using the parameters from previous frames. This is the main motivation for the proposed framework.

Since the ground plane pixels are removed from GPP, the footprints of objects can be simply segmented as connected components in binary GPP (after some simple filtering), e.g. Fig. 1(d). Hence our approach is a completely category free, unsupervised foreground object segmentation without using any color or other appearance information.

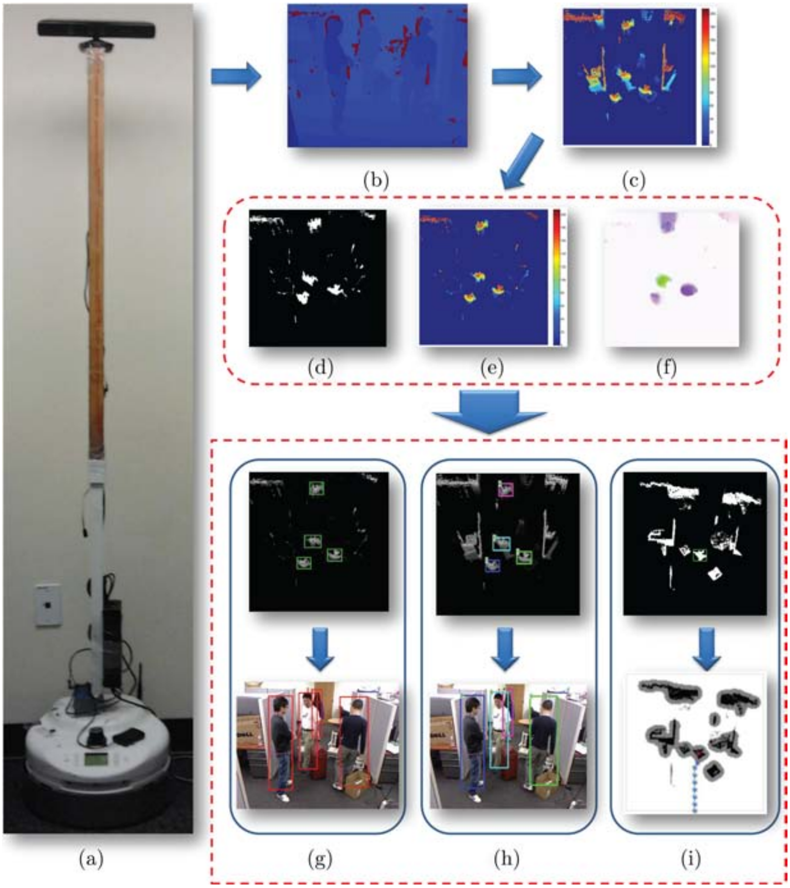


Fig. 1. Flow chart of our system. (a) Pekee II mobile robot equipped with a Kinect sensor. The ground plane projection (GPP) in (c) is obtained from the depth image in (b). (d) The foreground in GPP and their optical flow map. (e) The detected foreground objects in GPP. (f) Our multiple person tracking process in GPP. The tracked objects in GPP and their back projections to the RGB images are shown in (g) and (h). (i) Illustrates robot navigation process in GPP.

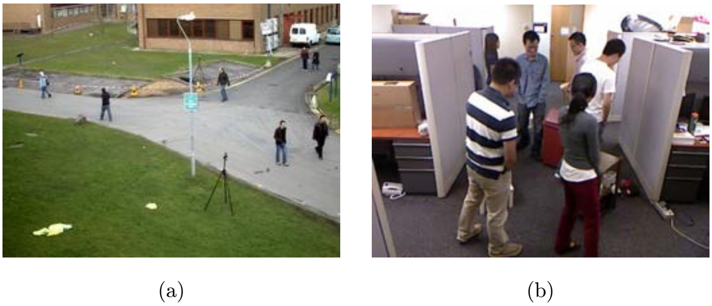


Fig. 2. Indoor Environment versus Outdoor environment. (a) Outdoor environment with multiple targets from PETS dataset. (b) Indoor environment with multiple targets from our dataset.

Although ground plan is commonly utilized for object location of mobile robots equipped with stereo cameras, the objects need to be first detected in the side-view (RGB) images in conventional methods.[9,25] The goal of this paper is not to present a novel GPP. In contrast, we aim to present a novel framework to utilize GPP. The key difference in our approach is that objects are first detected in GPP directly and then back projected to the side-view (RGB) images. This difference leads to nearly perfect detection results in the original RGB images in our system, with very low false negative and false positive rates.

Based on the footprints detection results in binary GPP, a novel PLA algorithm is proposed to link the detected target in consecutive gray GPP frames, which is the main contribution of our approach. Conversional multiple targets tracking methods use instance level association, where object detection is followed by a linking process. Hence when object detection fails, the linking process is likely to make mistakes, and the tracking process may fail. In contrast, in our approach, instance association is performed in gray GPP as PLA, where pixels are linked based on their optical flow (also computed in gray GPP). Here, the critical observation is that height value of a pixel in gray GPP is almost identical to the value in the previous frame due to high frame rate. Hence it naturally satisfies the brightness constancy property.[32] A flow map obtained from consecutive gray GPP frames is shown in Fig. 1(f). Since targets are clearly separated in GPP, we can easily track multiple moving targets in the cluttered scene.

We should mention that the existing online tracking algorithms[4,23,37] have great difficulty handling tracking multiple targets. "Online" means that the information from forthcoming frames cannot be used in current frame, only the information from previous frames can be used. On the other hand, existing multiple targets tracking algorithms[39,41] use the information from the whole sequence, and hence it may not be possible to extend them to "online" applications. The proposed approach performs online tracking and its implicity makes it possible to run it in real time, in particular, since our approach only uses pixel level cues in consecutive frames for linking.

With the inexpensive Kinect sensor, it is easy to extend our system to other applications like video surveillance in indoor environment, human–computer interaction, and robot vision. We utilize our approach for real-time robot navigation. While the navigation target is usually characterized as a static location on the map, many real life applications require the robot to navigate towards a specified moving target. For this kind of tasks, the robot needs to continuously identify and localize the target. We focus on such tasks and perform navigation by tracking in our framework. Hence the moving target is detected based on the footprints segmentation in binary GPP and tracked based on the PLA in gray GPP. Finally, a constrained A^* searching is utilized for path planning in GPP.

As we will demonstrate in our experimental results, the proposed approach yields robust and stable object detection results in both world coordinates of GPPs and in the original, RGB images, and the multiple targets tracking can achieve perfect performance even in very challenging environments. For robot navigation, our system is able to track moving targets successfully, which leads to correct navigation results.

Furthermore, we also consider dark indoor environments so that it is even impossible for humans to see anything in the side-view (RGB) images. With the help of Kinect sensor, our framework still works fine in such conditions as we will demonstrate in our results for detection, tracking, and robot navigation. This is a distinctive feature of our system, made possible by performing detection, tracking, and navigation in GPP. It is very useful for special environments like industrial and military environment where the normal lighting condition maybe unavailable.

The main contributions of the proposed approach are summarized as follow:

- A novel tracking framework based on GPP image. It not only allows for tracking in world coordinated, but also for robust tracking in side-view (RGB) images, by back projecting the tracked objects to them.
- A novel PLA algorithm is proposed for multiple targets tracking, PLA works on gray GPP and it works extremely well with both a static camera and a moving camera. It utilizes optical flow in gray GPP, which is a novel and extremely useful application of optical flow.
- The "online" and real-time computation speed makes our approach suitable for many challenging applications, in particular, since it works under normal light condition as well as in darkness.

The paper is organized as follows: Sec. 2 reviews the related work. In Sec. 3, GPP is introduced. The proposed detection, tracking and robot navigation algorithms are described in Secs. 4–6, respectively. Experiments are carried out in Sec. 7.

2. Related Work

The plan view-like representations are common for mobile robots.[2,21,38] Ess *et al.*[16] present a stereo-based system for the creation of dynamic obstacle maps for automotive or mobile robotics platform. There, pedestrians are first detected in 2D RGB images using a standard appearance-based detector. Then the position of detected objects on the ground plane is computed. This approach is then utilized to improve human tracking results[14] and in obstacle detection in crowded scenes.[10,15,34] In summary, those methods follow the traditional steps: first, detect objects in side-view (RGB) images by using appearance and color, and then find their locations on the ground plane. Hence these methods are sensitive to the detection results on side-view (RGB) images, the robot will not see the object if the 2D image detector cannot find the object in the side-view (RGB) image. As stated in the introduction, our framework is very different, since object detection and tracking are both performed in GPP image. If needed, the detection and tracking results are back projected to the RGB images. Our approach can achieve excellent detection and tracking results. Moreover, it can also be used to improve the performance of the above mentioned methods. Li *et al.*[31] demonstrated the fact that GPP makes object detection easier and more robust, even in cluttered scenes. Burschka *et al.*[9] propose a plan-view based obstacle detection and avoidance system for mobile vehicles equipped with a stereo camera. Their work includes ground plane estimation, plane removal, and grouping. Although their approach provides a vision-based alternative to the range sensors for robots, they only focus on detection of obstacles in plan view images. As we will show, GPP is also able to play an important role in object detection and tracking in the original color images. Harville and Li[25] describes a person tracking and activity recognition method that utilizes the maximum height of objects above the ground plane, which is represented as 2D height image. However, it does not back project the objects detected on the ground plane to the original images, i.e. it does not consider the relationship between GPPs and original images. As we show, the interaction between GPP and the side-view RGB image is a powerful tool to improve the detection and tracking.

Object detection is an intensely studied topic in computer vision, and there exists many kinds of detection strategies, like appearance-based detection paradigm.[13,19] In appearance-based detection, SIFT[33] or HOG[12] features are often used to capture the appearance information of the target. Shape information is also commonly used in detection.[6,40,48] All those methods work only with RGB images, and the main difference of the proposed approach is that we do not perform object detection by analyzing the original RGB image, instead, we perform detection as object footprint detection in the GPP image. Another difference is that all the detection methods mentioned above need to train a target model for a specific class, but our method is unsupervised and category free. This makes our algorithm easy to use for different applications.

Visual tracking is known as online category free tracking.[3,17,18,29,30,35,44,46,47] In this tracking paradigm, the target is manually selected in the first frame. Then the tracker will online track the target. Those methods use different strategies to learn the appearance model during tracking progresses. Our approach can also online track the targets, however, we do not need any manual labeling of the target in the first frame, since it detects moving targets automatically. Furthermore, our approach can easily track multiple targets, which is beyond the ability of most traditional visual tracking algorithms.

State-of-the-art approaches for multiple target tracking[7,22,36,42,43] do not need any initialization and can handle multiple targets naturally. These approaches are known as association-based trackers. The key difference between our approach and those multiple targets tracking methods is that our method is an online algorithm, while all those methods are offline algorithms, where multiple targets tracking problem is formulated as global optimization task. Another difference is that those multiple targets tracking methods need to train appearance model for specific class, hence

Table 1. Comparison of our framework with association-based tracking paradigm and category free tracking paradigm.

	Our Framework	Association-Based Tracking	Category Free Tracking
Target category	Category free	Specific category	Category free
Initialization	Auto, perfect	Auto, imperfect	Manual, perfect
Track solution	Individual	Global	Individual
Online/Offline	Online	Offline	Online
Motion Cue	Consecutive frames	Entire sequence	Consecutive frames

they are sensitive to the detection results in RGB images. A comparison of our tracking framework with other tracking methods is given in Table 1.

3. Ground Plane Projection

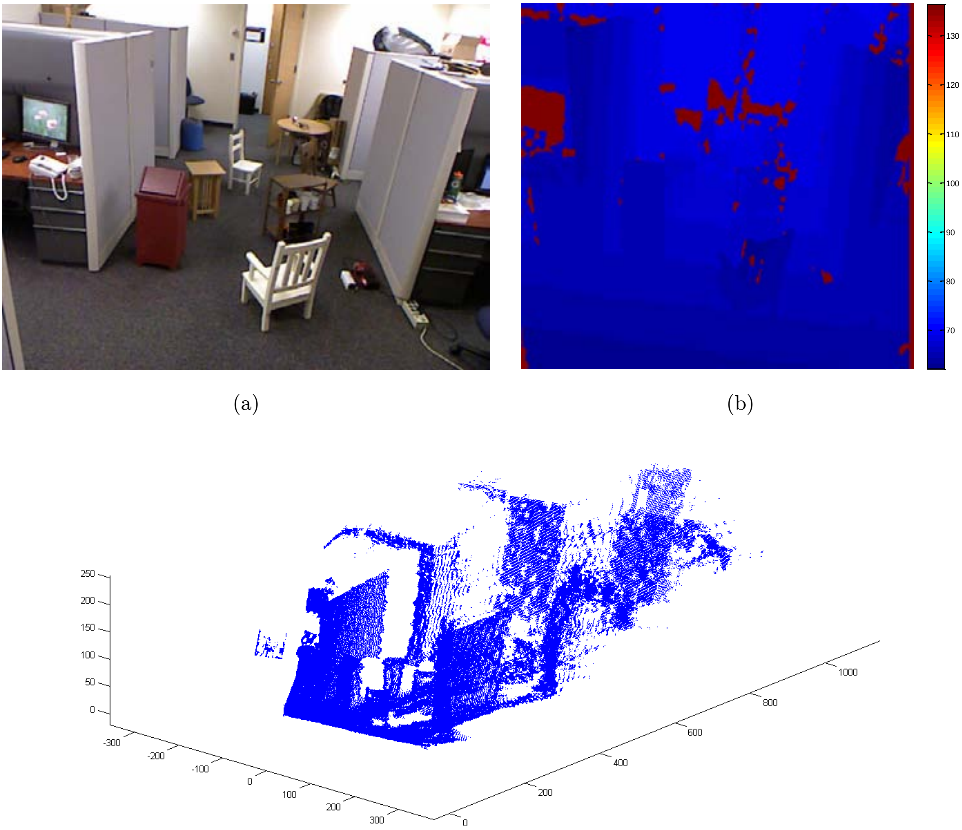
3.1. *Generating GPPs*

Given are a RGB frame I_t and a corresponding depth frame D_t at time t obtained with a Kinect sensor, as shown in Figs. 3(a) and 3(b), respectively. We first obtain the set of 3D points $\mathcal{K}_t$ from the depth map D_t, which represents a rough 3D scene reconstruction result, Fig. 3(c). The first step of our system is to find the ground plane $\mathbb{P}_t$ in $\mathcal{K}_t$. We utilize a RANSAC algorithm[20] to estimate it. We assume that the largest plane (measured in point count) such that there is no other points below it is the ground plane $\mathbb{P}_t$. This assumption is usually satisfied if the camera is looking down, and the floor, which is usually the ground plane in indoor applications is not too cluttered. This assumption can be weakened if rough estimates of roll and pitch angles of the camera relative to the ground plane and of the camera height are known. Then the ground plane is the largest plane with no other points below it within the limits set by the rough estimates.

We denote with $\mathcal{P}_t$ the set of 3D points that lie on the ground plane $\mathbb{P}_t$. (The RANSAC plane fitting algorithm returns both the plane equation and the set of its points.) Then we perform coordinate change transformation T from the original camera coordinates of points $\mathcal{K}_t$ to new coordinates $T(\mathcal{K}_t)$ such that the ground plane is the (x, y)-plane and the camera (x, y, z) coordinates are $(0, 0, h)$, where h is the camera height (above the ground plane). We also observe that the z coordinate of each point in $T(\mathcal{K}_t)$ denotes its height (above the ground plane).

After removing the ground plane points $T(\mathcal{P}_t)$, we project the remaining points to the plane $T(P_t)$, i.e. to (x, y)-plane. More precise, we project points in $T(\mathcal{K}_t \backslash \mathcal{P}_t)$ to plane $T(P_t)$ by simply dropping their z coordinates. We obtain the GPP image H_t by first quantizing plane $T(P_t)$ to a square grid with the square size of $1\,\text{cm} \times 1\,\text{cm}$ and then counting the number of points projected to each square. The value of a pixel of H_t, representing a square, is set to the maximum height of pixels projecting to the corresponding square. Hence H_t is a gray level image.

A simple filter operation is performed to achieve a binary GPP image G_t. A pixel of G_t representing a square is set to 1 if there is more than k 3D points projecting to



(a)

(b)

(c)

Fig. 3. Scene layout reconstruction. (a) Side-view (RGB) frame, (b) depth frame and (c) the scene layout (3D point cloud).

it, and it is set to 0 otherwise. The 0 values in G_t indicate vacant space, while the 1 values represent occupied space. The threshold k, which is set to 12 in all our experiments, allows us to eliminate noisy and outlier 3D points. The threshold on the count of 3D points acts as a low-pass filter that removes such noisy points. In particular, it efficiently eliminates points on the floor that are incorrectly recovered as points above the floor. This is important to eliminate phantom objects. H_t and G_t are both important for our approach. In particular, G_t is very useful for object detection in GPP while H_t for motion estimation.

3.2. *From GPP back to the original image*

It is easy to determine the "back projection" from H_t or G_t to the depth frame D_t at time t. We know the correspondence between pixels in D_t and 3D points cloud $\mathcal{K}_t$. We also know the correspondence between 3D points cloud $\mathcal{K}_t$ and pixels in H_t or G_t.

Hence we can define a mapping π from D_t to H_t. The "back projection" is the inverse mapping π^{-1}. We can also interpret π^{-1} as the mapping from H_t to I_t (the RGB frame at time t), since I_t and D_t can be calibrated so that the correspondence of their pixels is known.

4. Multiple Object Detection

4.1. *Object detection with footprint segmentation in GPP*

There are several reasons why the task of object detection in GPP is greatly simplified as compared to the original images. There is no perspective distortion in GPPs and the scale is known. Moreover, the impact of occlusion that is typical to perspective (or orthographic) projection in regular 2D images is significantly reduced in GPPs.

Since the GPP frame G_t is a binary image, we can simply use a connected components labeling algorithm as footprint detector in G_t. This simple process allows us to detect all the targets supported by the ground plane, but the disparity maps usually only provide good information at the edges, and consequently GPPs may contain object footprints that are incomplete (in particular, objects farther away from camera). As Fig. 1(c) shows, there exist lots of invalid pixels (holes) in GPPs. To reduce the invalid pixels, a two-step preprocessing is applied to fill the vacant pixels:

(1) A value of each 0 pixel is replaced with the value of its nearest neighbor in a 4×4 window.
(2) A median filter with 3×3 window is used to smooth GPP images.

Although this preprocessing substantially improves the quality of GPPs, it cannot fill larger gaps. Therefore, we enclose the detected connected components with bounding boxes. We also utilize a prior on the expected size of the bounding boxes. Bounding boxes which are too large or strip-like are not considered, for they are probably background clutter or walls. This allows us to focus on foreground objects and ignore the background. A representative set of detected footprints and the corresponding back projected objects is shown in Fig. 1(g). The upper image shows our footprint detection results in GPP image H_t, the lower image shows the "back projected" objects on the side-view (RGB) image.

4.2. *Background model*

Our footprint detection detects all objects on the ground plane, which include both targets and obstacles, but we only interested in the moving targets like persons in this work. On the other hand, the depth information from Kinect is sometimes unstable, which makes some static objects in GPP look like moving objects. To focus

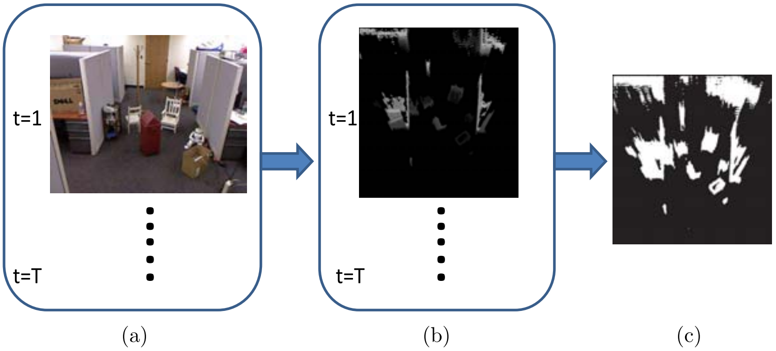


Fig. 4. Background model of a GPP video (a) are the original RGB images up to time T, (b) are the GPP frames of the scene, and (c) is the background model learned from the GPP frames.

on moving targets like persons, a background model is learned as follows:

$$\mathrm{B}(x,y) = \begin{cases} 1 & \text{if } H_t(x,y) > \beta,\ t \in \{1,\ldots,T\}, \\ 0 & \text{else,} \end{cases} \tag{1}$$

where H_t is the gray level GPP frame at time t, e.g. Fig. 4(b), T is the total number of video frames used to build the background model, and β is a threshold. With this simple background learning, we are able to obtain a stable background model, e.g. Fig. 4(c).

In order to detect all the moving target on H_t, we remove the scene noise by pixel wise multiplication between GPP frame H_{t+1} and B:

$$\mathrm{H}_{t+1} = H_{t+1} * \mathrm{B}, \tag{2}$$

where H_{t+1} is the GPP image after removing the scene noise, e.g. Fig. 1(e), the moving targets in H_{t+1} are clear enough compared with H_{t+1}, e.g. Fig. 1(c). We also generate the binary image G_{t+1} based on H_{t+1}, e.g. Fig. 1(d). Then it is easy to locate multiple moving targets in G_{t+1} using only the footprint detection described above.

5. Multiple Moving Target Tracking with PLA

In this section, we present a robust and efficient multiple target tracking algorithm based on GPP. In our system we do not need to learn any model of the target nor manually label the target to initialize the tracking, since all entering objects could be robustly detected in G_{t+1} as described in Sec. 4.

We formulate the multiple target tracking as index ID assignment task. Suppose we have M targets $\{O_{(1,t)},\ldots,O_{(M,t)}\}$ in frame H_t at time t, each has an numerical index ID $\mathcal{L} \in \{1,\ldots,L\}$, where L is the total number of different targets that appear in all the previous frames. L may not be equal to M at time t. As described in Sec. 4, we can detect the candidates set in frame H_{t+1} at time $t+1$, i.e. N connected components denoted as $\{O_{(1,t+1)},\ldots,O_{(N,t+1)}\}$ are detected. Then we assign an index ID $\mathcal{L}$ to each pixel inside the candidate set. The probability of $O_{(n,t+1)}$ being assigned an index ID is defined as

$$P(O_{(n,t+1)}, \mathcal{L}) = \frac{\#\{L(p) = \mathcal{L}\}}{R}, \tag{3}$$

where # is the count operation, R denotes the total pixel number inside the connected component $O_{(n,t+1)}$, and $L(\cdot)$ indicates the index ID of a pixel at the spatio-temporal position $p = (x, y, t)$ in H_t, it is defined in Sec. 5.2.

Finally, the index ID with the highest probability is selected, i.e. the index ID function $\mathbf{L}$ for $O_{(n,t+1)}$ is given by

$$\mathbf{L}(O_{n,t+1}) = \arg\max P(O_{(n,t+1)}, \mathcal{L}). \tag{4}$$

Here we enforce a hard constrain that $\mathcal{L}$ can only assign each index to at most one candidate. If a target is in a start state, we assign a new index $(M + 1)$ for it.

5.1. *State prior*

In our system, the objects can be detected automatically when they appear in the scene. We only have two simple states prior for the index ID, called *Start State* and *Stop State*. Such states prior are not a special assumption for our system, but they belong to common-sense assumptions used in the multiple-target tracking literatures, e.g. Refs. 7, 22 and 36.

Start State: A new index ID will be assigned to a new target appearing from the boundary of the scene. If a target appears near the scene center, then it might not be a new target, but the background noise. This common-sense prior makes our system robust to phantom regions.

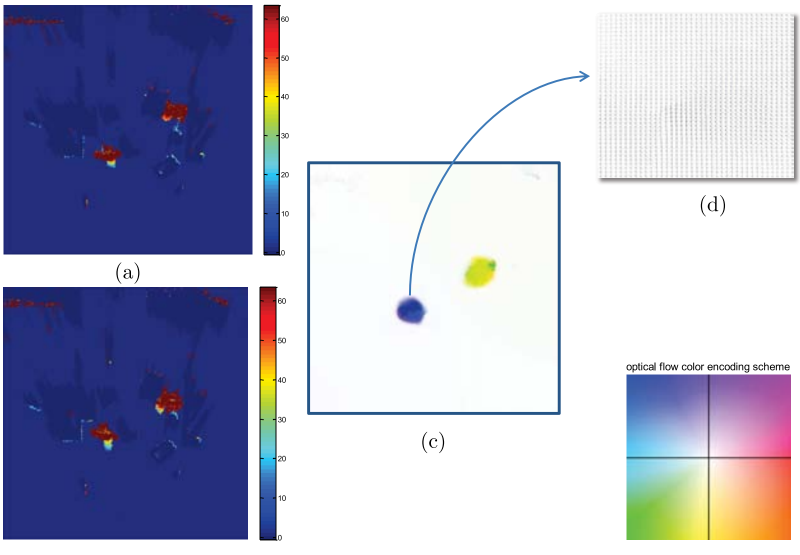
Stop State: A index ID will not be assigned to any target if the ID holder target leaves the scene. If the same target comes back later, we will still assign a new index to it.

Another special case we should mention is that if there exist multiple targets in the first frame, we will randomly assign an index for each target. This does not contradict the *Start State* assumption, because it is possible that many targets exist at the center area in first frame.

5.2. *PLA in GPP*

The index assignment task in multiple target tracking is also known as instance linking task, where an instance in the previous frame is linked to an instance in the current frame if they are the same target. We follow the linking paradigm, but decompose the traditional instance linking task into PLA. The whole process is illustrated in Fig. 5, where the input is two consecutive GPP frames H_t and H_{t+1}.

We recall that each pixel in GPP image H_t represents the maximal height above the ground plane of 3D points projecting to this pixel. We observe that the height value of a pixel in H_t is almost identical to the value in the previous frame due to high



(a)

(b)

(c)

(d)

(e)

Fig. 5. PLA. (a) and (b) are consecutive GPP frames; (c) is the optical flow map obtained from (a) and (b); (d) shows optical flow vectors of the selected region in (c); (e) is the optical flow direction chart color encoded.

frame rate. Hence it naturally satisfies the *Brightness Constancy* property[5]:

$$H_t(x, y) = H_{t+1}(x + u, y + v), \tag{5}$$

where (x, y) represents pixel coordinates in H_t and (u, v) are the horizontal and vertical components of the flow field.

Let the spatiotemporal position of a pixel be denoted with $p = (x, y, t)$. The PLA is expressed as the estimation for (du, dv) which denotes the motion of p. We follow the incremental flow framework. Let the flow field be known as $w = (u, v, 1)$. The objective is to estimate the best increment $dw = (du, dv)$. We use the Iterative Reweighed Least Squares (IRLS)[32] to obtain the solution optical flow vectors (du, dv). Figure 5(c) shows the optical flow map obtained from (a) and (b), where the direction of the optical flow vectors is color coded according to the chart in (e). Figure 5(d) shows the vectors of the selected region in (c).

Based on (du, dv), we can easily find a pair of corresponding pixels $p(x, y, t)$ and $p'(x', y', t + 1)$ in consecutive frames H_t and H_{t+1}. For $p(x, y, t)$ belongs to a connected components in H_t, it has a index $\mathcal{L}$. We propagate it to $p'(x', y', t + 1)$:

$$L(p'(x', y', t + 1)) = L(p(x, y, t)), \tag{6}$$

where $x' = x + du$ and $y' = y + dv$.

Finally, Eqs. (3) and (4) are used to assign labels to objects, i.e. each connected component is assigned the label of the majority of its pixels.

5.3. *Splitting and merging connected components*

Based on PLA, split and merging operation are employed to handle multiple targets cross walking and partial/full "occlusion" in GPP. Occlusion can still happen in GPP, since we perform online tracking, and consequently, use only a single view at each time step t.

Targets cross walking often lead to the index ID switch, as shown in Fig. 6(b). This is also a problem in the GPP, since both targets are labeled as a single connected component as shown in (c). However, the optical flow in GPP provides a useful tool for solving this problem. The optical flow estimation is shown in Fig. 6(d). Obviously, we can split the two targets for they have very different flow. Hence we utilize the optical flow information to split connected components in GPP.

As mentioned above, although occlusion in GPP is less likely than in the side-view (RGB) images, it is possible, since GPP construction is based on a single depth image and a target can be colluded or partially occluded in the depth map. This leads to missing or incomplete height information in GPP H_t. We call this phenomenon as "*occlusion in GPP*", e.g. in Fig. 6(f), the target is partially occluded by a chair, which splits its connected component into many parts in Fig. 6(g).

The proposed PLA can handle such condition naturally, since in the previous frame, shown in Fig. 6(e), the target is not occluded, and based on our PLA shown in Fig. 6(h), the split connected components can be merged. The merged component will be assigned the same index, because at most one index can be assigned to one target. Hence PLA in GPP provides us with simple but very robust means for handling partially occluded targets.

To handle the long time fully occluded targets in GPP, we utilize the following criterion. For a target candidate $O_{(n,t+1)}$ at certain location, its index has strong association with indexes of spatially nearby targets in previous several frames. We consider all the indexed targets O_N (N means neighbors) within a small temporal window, and compute the probability of assigning index $\mathcal{L}$ to $O_{(j,t+1)}$, which can be stated as:

$$P(O_{(j,t+1)}) = \exp(-D(\alpha_n, \alpha_{(j,t+1)})), \tag{7}$$

where $\alpha_n, n \in \{1, \ldots, N\}$ is the location of target O_n in the previous frames, and $\alpha_{(j,t+1)}$ is the location of target O_j in current frame. D is Euclidean distance. While location information can successfully separate target candidates that are far away from each other, there is a serious danger that the indices of target candidates will be wrongly switched when they are relatively close.

Optical flow seems to be effective for low level motion estimation, but if the pixel do not move in consecutive frames, e.g. a person stay in the same location in several frames, then the optical flow cannot be used. In this case, the flow map will appear white color as shown in Fig. 5(e), the center region. Again our PLA can handle this case naturally. This is so, since the detected regions (connected components) in

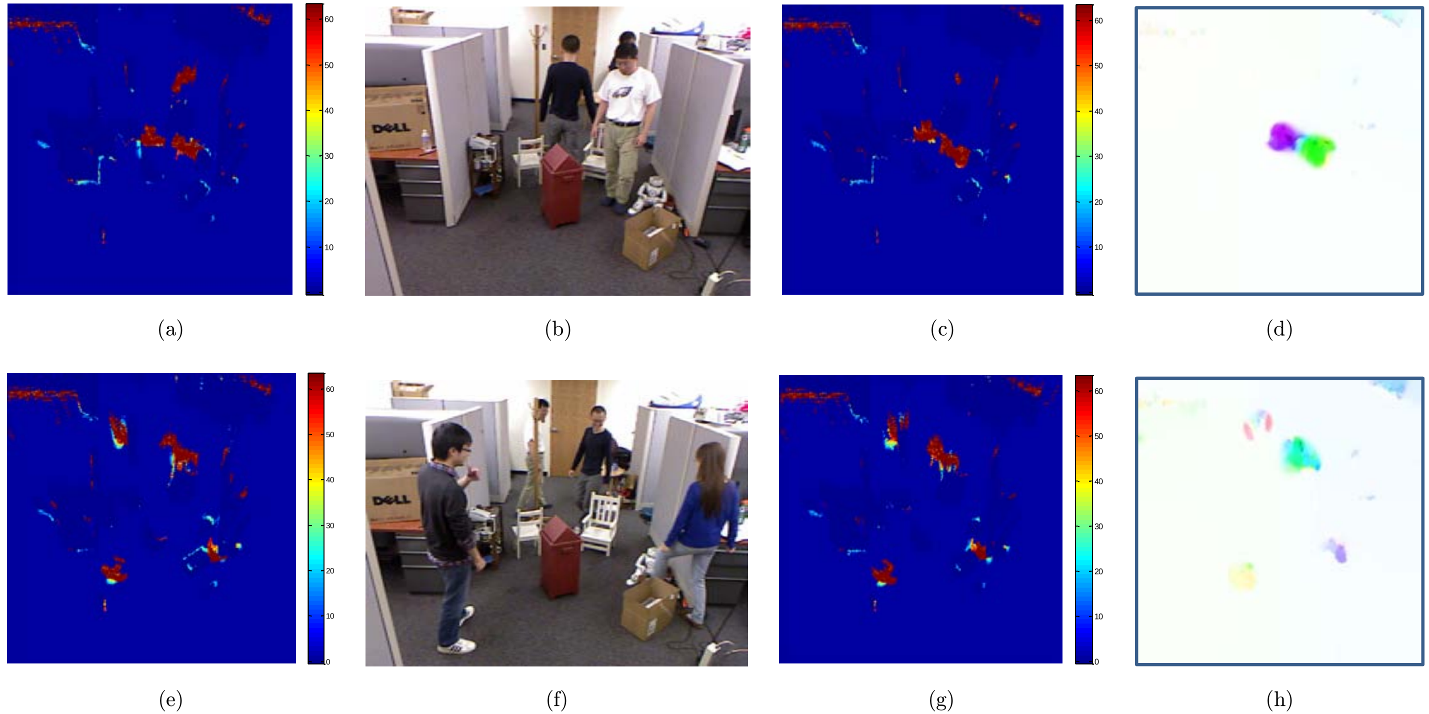


Fig. 6. First row illustrates splitting a connected component with two targets. (a) and (c) are consecutive GPP frames and (b) is the RGB image corresponding to (c). (d) shows the optical flow map from (a) and (b), which is used for splitting. Second row illustrates merging several components of a single target. (e) and (g) are consecutive GPP frames and (f) is the RGB image corresponding to (g). (h) shows the optical flow map from (e) and (f), which is used for merging.

binary GPP will at the same location in consecutive frames. Consequently, there is no doubt that we should link the candidates at the same location even the optical flow vectors are close to zero in this region.

6. Robot Navigation

In this section, we extend out multiple targets detection and tracking framework to robot navigation in cluttered indoor environments. Our navigation framework is implemented on a Pekee II mobile robot shown in Fig. 1(a). It has a round base with a tall extension (approximately 200 cm). The robot is equipped with Kinect sensor, which can also be replaced with a BumbleBee2 stereo camera.[28]

The pipeline of our navigation framework is illustrated in Fig. 7. After the target object is manually selected in the first frame, the robot approaches it through a series of small movements in the perception/action cycle. In each cycle, the robot first acquires RGB-D frame. Next, it estimates the ground plane $\mathbb{P}_t$ and the GPP images H_t and G_t as we described in Sec. 3. Here, binary GPP image G_t is used to indicate free space for robot motion and space occupied by obstacles. We should claim that we do not learn any background model for robot navigation, this is different from Sec. 3. However, our PLA algorithm still can identify and track the target successfully.

Next, A* path planning[24] with constraints is used to plan a path toward the target object. Finally, the robot executes the first segment of the path only and turns to face the target. Then the whole procedure repeats.

6.1. *Target identification*

To identify the navigation target in consecutive frames, an extension of our multiple target detection and tracking algorithm is proposed in this section. Same with the detection and tracking, our identification of the specified target is also executed on GPP images. The difference is that we do not need any background model.

Suppose we know the specified target O_t in frame H_t at time t (shown in Fig. 8(d)). To identify the target in $H_{(t+1)}$, the detection algorithm described in Sec. 4 is used to generate a candidate set, e.g. $\{O_{(1,t+1)}, \ldots, O_{(N,t+1)}\}$ (shown in Fig. 8(e)).

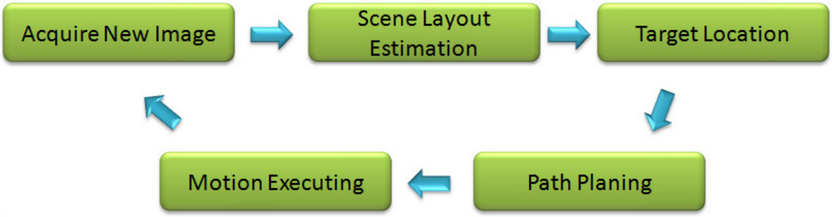
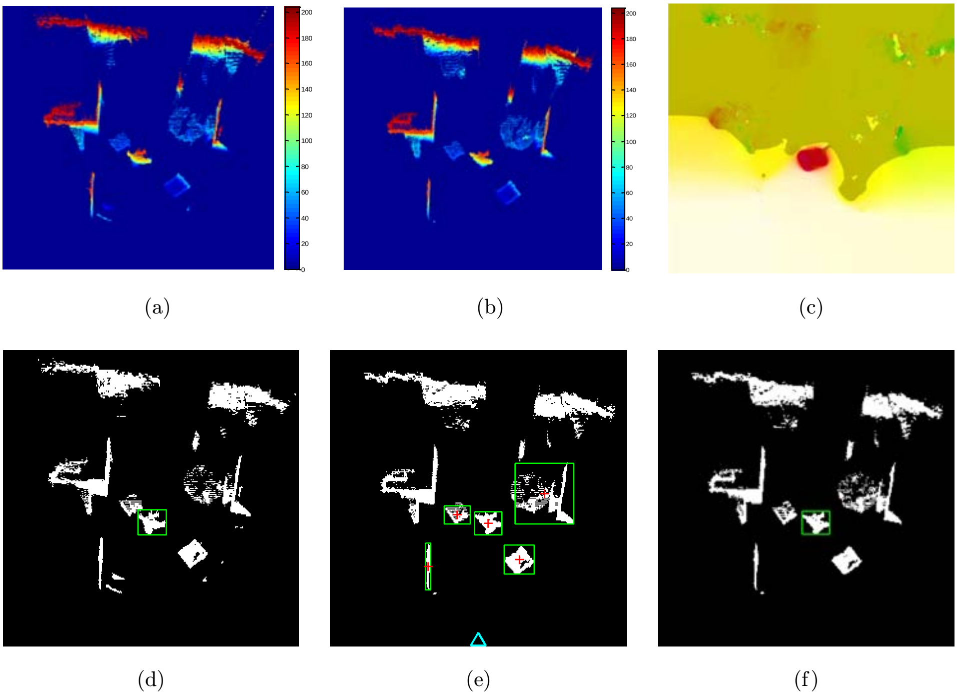


Fig. 7. Navigation system pipeline.



(a) (b) (c)

(d) (e) (f)

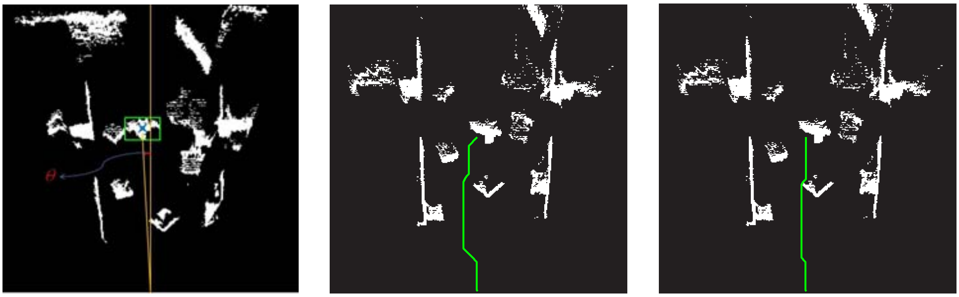
Fig. 8. Target tracking from mobile platform using optical flow in GPP. Two consecutive GPP images are shown as H_t, H_{t+1} in (a,b) and as G_t, G_{t+1} in (d,e). The green bounding box marks the tracked target in (d). The candidate footprints are marked with green rectangles in (e). (c) Shows the optical flow field obtained from H_t and H_{t+1}. It is used to link the target to correct candidate at time $t+1$ as shown by the green bounding box in (f).

The modification is that we do not need to remove the background noise, hence the candidate could be the moving target or the obstacle. We also formulate the identification of the target task as index assignment task as described in Sec. 5. In the navigation case, only the specified target is treated as having an index ID. Then the PLA algorithm is used to propagate the index from the target to the candidates. Based on Eq. (6), only the real target $O_{(t+1)}$ at time $(t+1)$ can obtain the index from the specified target at time t, hence it is easy to identify and relocate the specified target in frame H_{t+1} at time $(t+1)$.

As shown Fig. 8, (a) is H_t and (c) is $H_{(t+1)}$, which are both plotted in color space. (e) is the estimated flow map based on (a) and (c). For the color in the flow map encodes the motion of the pixel between H_t and H_{t+1}, it is clear to see that the motion of the specified target region is very different from the surrounding region.

6.2. *Constrained A^* algorithm*

To plan the path for the mobile robot, the basic A^* algorithm[24] with two constraints is utilized in our framework.



(a) (b) (c)

Fig. 9. The influence of our constraints on A^* algorithm. (a) The turning angle needed for the robot to face the target. (b) A^* path with distance transform term as used in our framework. (c) A^* path without the distance transform.

Constraint 1: The robot should always face to the target, we implement this constraint by calculating the angle between the robot and the target, and controlling the robot to turn the angle before path planning. This constraint is used to guarantee the target is always inside the view of the robot, e.g. Fig. 9(a). The path planning is then performed based on the new view of the robot.

Constraint 2: The robot should avoid going too close to obstacles. We implemented this constraint by adding a distance transform term to the A^* cost function. This constraint is used to guarantee the robot will never collide with an obstacle in its path. As shown in Fig. 1(i), the black regions are occupied by object, the gray regions denote the constraint area that the robot could not walk, the white regions represent the free space. The comparison between the A^* algorithm with Constraint 2 and without Constraint 2 is shown in Figs. 9(b) and 9(c).

7. Experimental Results

In this section, we evaluate our framework on following tasks: multiple moving target detection, multiple moving target tracking, and autonomous navigation by tracking in cluttered indoor environments.

DataSet: For detection and tracking experiments, we fix the position of the mobile robot and use static Kinect camera to collect eight challenging test videos. The videos are collected at 5 FPS with the resolution of 640 × 480. In the test videos, up to six people are shown in different poses, with full and partial occlusion, large pose deformation, large scale variation, and motion with different speeds. There also exists six foreground obstacles in the videos. The annotations identify all humans present in the area of 3 m × 2 m on the ground plane, which is the area where the Kinect depth map readings can be obtained. The videos are summarized in Table 2.

We also collected two videos in dark indoor environments so that almost nothing can be seen in the side-view (RGB) images. These are sequences 09 and 10 in Table 2.

Table 2. Test videos.

	Frames	Max # Persons Per Frame	# Persons in All Frames
Seq.01	210	2	354
Seq.02	230	4	743
Seq.03	220	2	243
Seq.04	210	3	521
Seq.05	200	3	462
Seq.06	240	3	587
Seq.07	200	6	1017
Seq.08	220	6	1211
Seq.09	150	3	—
Seq.10	140	3	—

Since it is hard to annotate the RGB images in these videos, we only evaluate detection and tracking results in Sec. 7.2.

For robot navigation evaluation, we use the Pekee II robot equipped with Kinect sensor to track a specified person in different indoor environments with various obstacles and multiple person walking.

In all the experiments the Kinect camera is facing down, which is needed to estimate the ground plane in all images.

7.1. *Multiple targets tracking*

In this section, we focus on evaluating performance of tracking multiple moving persons. Experiments are carried out to validate the proposed approach presented in Sec. 5.

Quantitative Comparisons: We evaluate our tracking performance with CLEAR Metrics.[8] We use four metrics: Multiple Object Tracking Precision (MOTP), False Positive Rate (FPR), Miss Rate (MR), Number of Miss Match (IDs) and Multiple Object Tracking Accuracy (MOTA). We give a short description for those evaluation methodologies in the following.

MOTP is the total error in estimated location for matched object-hypothesis pairs over all frames, averaged by the total number of matches made, which can be represented as MOTP $= \sum_{i,t} d_t^i / \sum_t c_t$, where d_t^i denotes the distance between the object-hypothesis i and the ground truth, c_t indicates the number of matches made in time t.

MOTA is used to account all object configuration errors made by the tracer and is defined as

$$\text{MOTA} = 1 - \sum_t (m_t + fp_t + mme_t) \Big/ \sum\nolimits_t g_t,$$

where m_t, fp_t and mme_t are the number of misses, false positives, mismatches (ID switch) at time t, respectively.

$\overline{fp} = \sum_t fp_t / \sum_t g_t$ is the total FPR, $\bar{m} = \sum_t m_t / \sum_t g_t$ is the ratio of the misses (MR) in the sequence, where g_t is the number of targets present at time t. $\overline{mme} = \sum_t mme_t / \sum_t g_t$ is the mismatch rate (IDs).

The multiple targets tracking results are compared with the state-of-the-art multiple target tracking algorithms DP and its extension DP + NMS, which are both proposed in Ref. 36. Both DP and DP + NMS use detected targets in RGB images as the candidates for linking. LSVM[19] is used for human detection in RGB image as the preprocessing for DP and DP + NMS.

Table 3 report the performance measured with CLEAR Metrics. Our method significantly outperforms the other two algorithms. The reason is that the methods in Ref. 36 rely on the detection results in RGB images. Hence a high detection FPR and MR affect the tracking result greatly. Furthermore, Ref. 36 also have an assumption that appearance and trajectory should not change greatly. It is well known that this assumption is not adequate for challenging indoor environments.

Since our PLA works on consecutive GPP frames and since the targets are clearly separated, due to the removal of the background noise, the GPP motion cues are

Table 3. Performance evaluation in CLEAR metrics.

		MOTP	MR (%)	FPR (%)	IDs	MOTA (%)
Seq.01	**Our**	**15.48**	**4.52**	**0**	**0**	**95.48**
	DP + NMS	18.82	27.31	68.85	4	2.93
	DP	18.83	26.64	69.53	9	1.81
Seq.02	**Our**	**16.85**	**6.19**	**0.81**	**0**	**93.00**
	DP + NMS	31.16	43.07	49.26	7	6.64
	DP	31.15	41.15	51.18	9	6.34
Seq.03	**Our**	**19.10**	**8.23**	**0.82**	**0**	**90.95**
	DP + NMS	38.11	52.03	42.97	5	2.73
	DP	38.11	50.88	44.82	7	1.12
Seq.04	**Our**	**16.28**	**5.17**	**0.74**	**0**	**94.09**
	DP + NMS	26.11	41.30	53.99	3	3.29
	DP	26.15	39.98	54.63	4	3.49
Seq.05	**Our**	**21.19**	**4.73**	**1.07**	**0**	**94.20**
	DP + NMS	42.89	38.14	51.88	3	8.48
	DP	42.88	36.99	53.50	4	7.51
Seq.06	**Our**	**15.83**	**6.21**	**1.31**	**0**	**92.48**
	DP + NMS	35.66	31.33	55.35	4	11.66
	DP	35.67	30.90	56.97	4	10.47
Seq.07	**Our**	**17.61**	**8.72**	**4.31**	2	**86.02**
	DP + NMS	59.31	54.96	43.21	**1**	1.33
	DP	59.30	53.11	44.85	2	1.04
Seq.08	**Our**	**29.83**	**17.69**	**6.11**	5	**73.93**
	DP + NMS	51.11	69.11	30.13	**0**	0.76
	DP	51.13	68.42	30.89	**0**	0.69
Average	**Our**	**19.02**	**7.68**	**1.89**	**0.86**	**90.02**
	DP + NMS	37.89	44.65	49.45	3.38	4.73
	DP	37.90	43.51	50.79	4.88	4.06

clearly sufficient to link the different targets. Because the detection process on GPP gives good detection results, we have a very low MR as shown in Table 3. Only in the Seq. 08 the MR is a bit higher. The reason is that in this video, we always have about six persons in a limited indoor space. As shown in Fig. 10 (Seq.08(a)–(c)) this video is

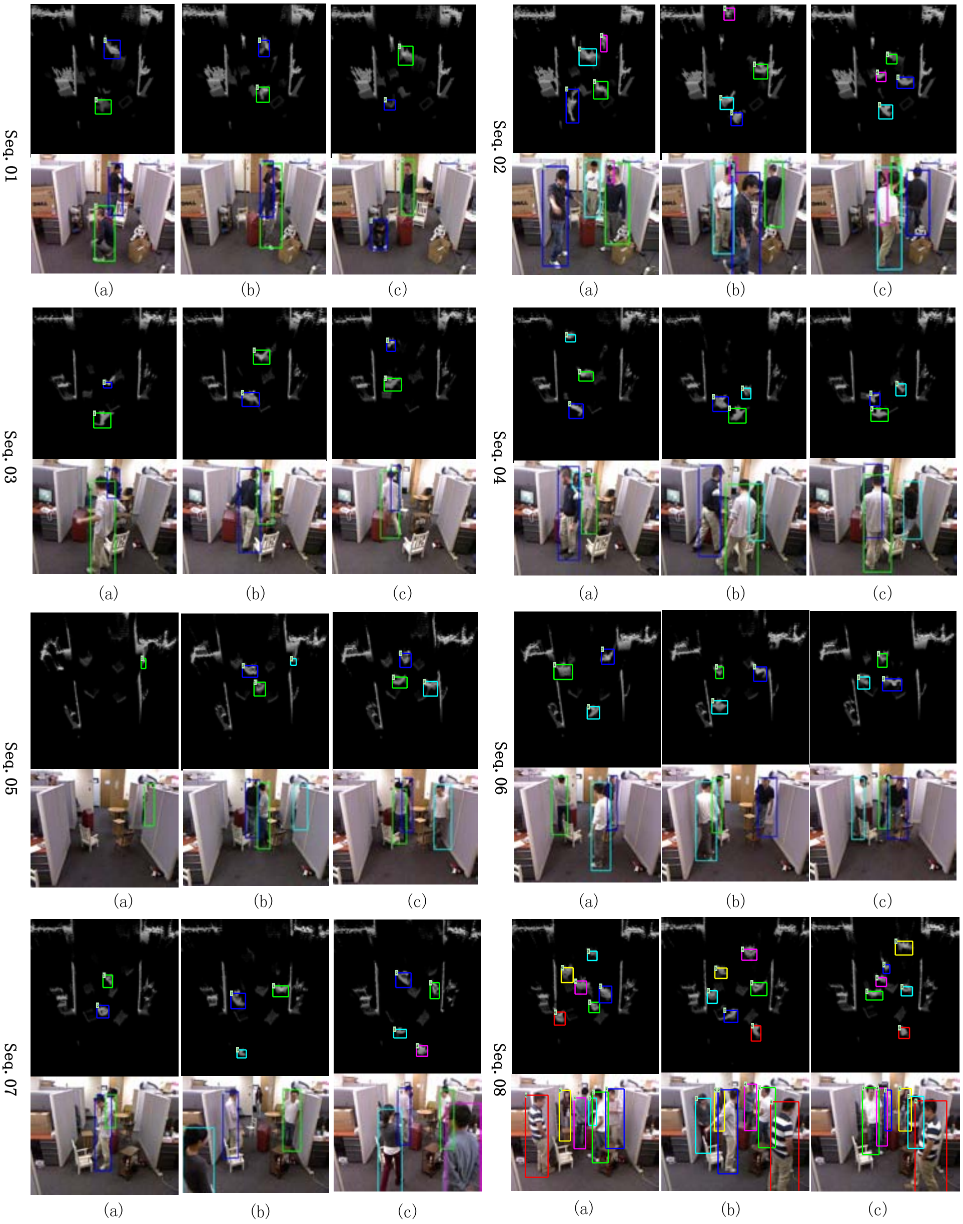


Fig. 10. The top row shows our tracking results in GPP frames, and the second row shows the tracking results back projected to the RGB frames. Bounding boxes with the colors represent the same target.

very challenging for the scene is very cluttered, occlusions exist in almost very frame. The detection and tracking is a few orders of magnitude more challenging in RGB images. As shown in Table 3, MR for Seq.08 of the two methods in Ref. 36 is 69.11% and 69.42%, which means that they fails to track most of the targets. In contrast our MR is 17.69%.

We have low FPR in all videos, which means that we successfully track most of the targets. Our method and the two methods in Ref. 36 have low IDs in all videos, but the reason is different, since the IDs are counted only in frames in which moving targets are detected. So our IDs make sense for we have a high success tracking rate, and low IDs mean that we can always separate the different targets. In contrast, Ref. 36 have a very a low tracking rate. The MOTA shown in Table 3 clearly show that our overall performance greatly outperforms the two methods in Ref. 36. Example tracking results are shown in Fig. 10.

7.2. *Multiple target detection and tracking in dark indoor environments*

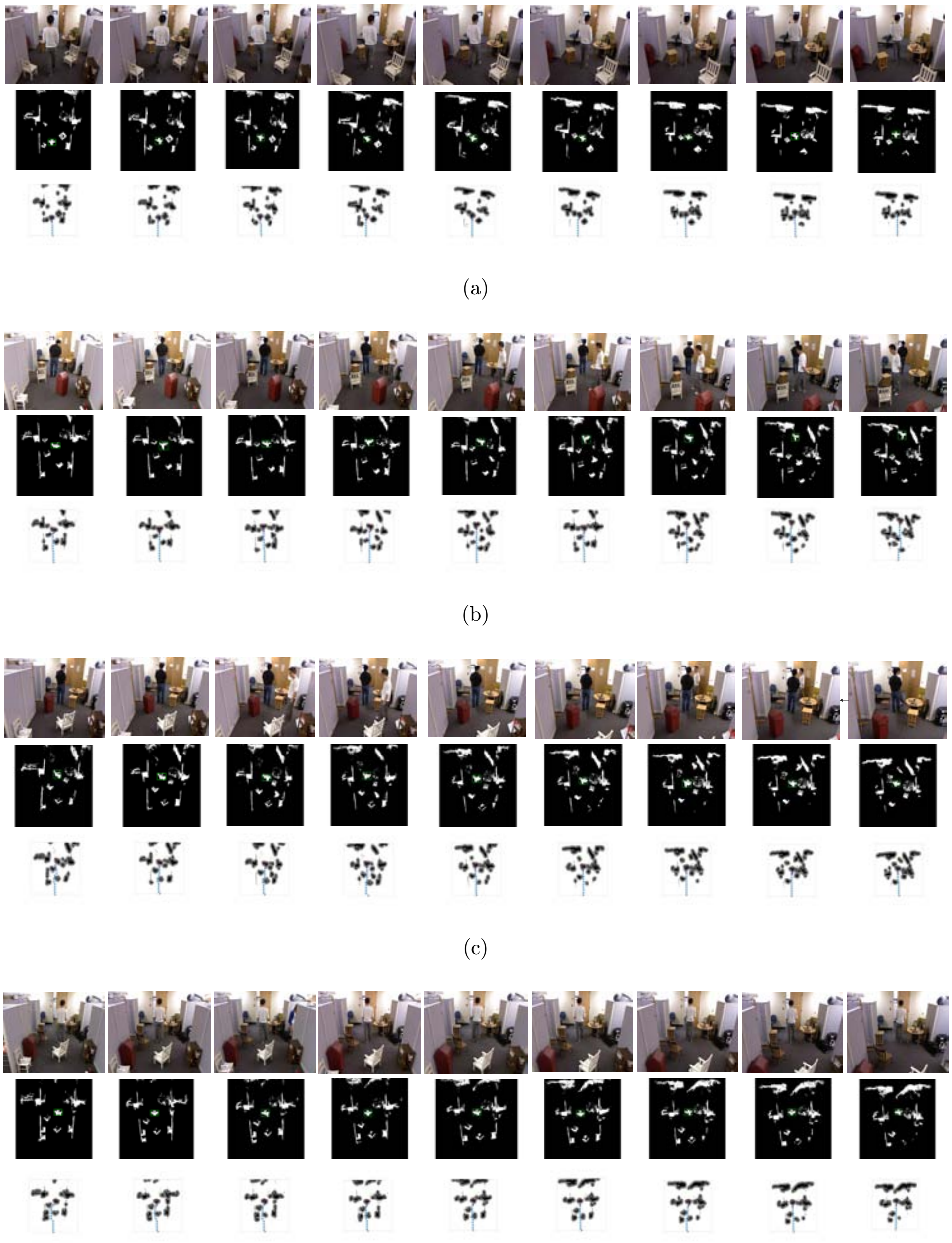
In this section, we test our framework in dark cluttered indoor environments. As shown in the first row in Fig. 12, Seq.09 and Seq.10, we almost cannot see anything in the RGB images. Hence the traditional detection and tracking algorithm cannot work on this case. Thanks to Kinect sensor, the depth information is not affected. Consequently, our tracking framework in GPP images still can be utilized, and its performance is unaffected. Figure 12 illustrates some of our detection and tracking results.

7.3. *Robot navigation*

In this section, we evaluate our method for real-time robot navigation by tracking. Our robot Pekee II is used. For navigation, we need the user to manually label the target we want to track in the first frame. Then the robot autonomously performs navigation by tracking as described in Sec. 6.

Figure 11 shows some navigation results. The obstacle locations are different in each video, and there are also other moving target that walk access in the environment as shown in Figs. 11(b) and 11(c). The tracking of the target is a very challenging task, as illustrated in Fig. 11. However, with the detection and tracking in GPP, our robot can always track the target person successfully, even in the presence of other moving persons in close proximity. We need to stress that no appearance information nor learning was utilized. The reason for the successful navigation by tracking is obvious: although in the RGB frames, the scene is very cluttered as shown in Fig. 11, the first row of (a)–(d), the target is clearly separated from the background in the GPP images.

We also illustrate robot navigation in dark indoor environments in Fig. 13, where first we have the normal lighting condition for the user to label the target, and then the robot navigation is done in dark environments. The proposed navigation by tracking method was also successful under such conditions.



(a)

(b)

(c)

(d)

Fig. 11. The first row of each sequence shows the side-view RGB images. The second row shows the GPP images and the green box indicating the tracked target. The third row of each sequence shows the path planning results in GPP (color online).

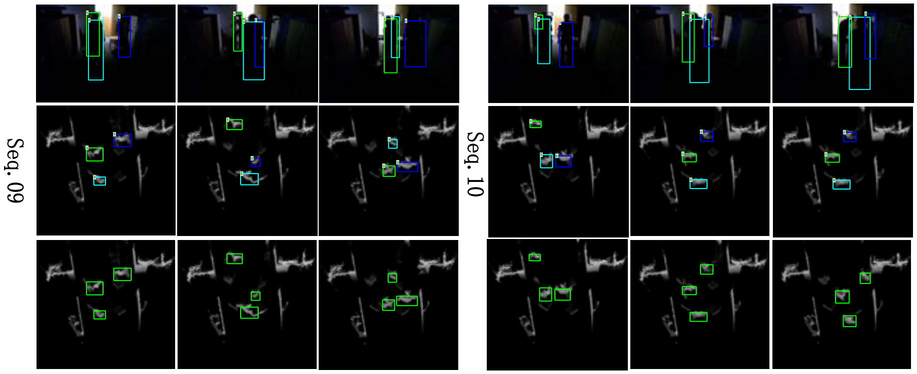


Fig. 12. Detection and tracking results in dark indoor environments. The first rows show the back projected tracking results on RGB images. The second rows show the tracking results in GPP images. The third rows show the detection results in GPP images.

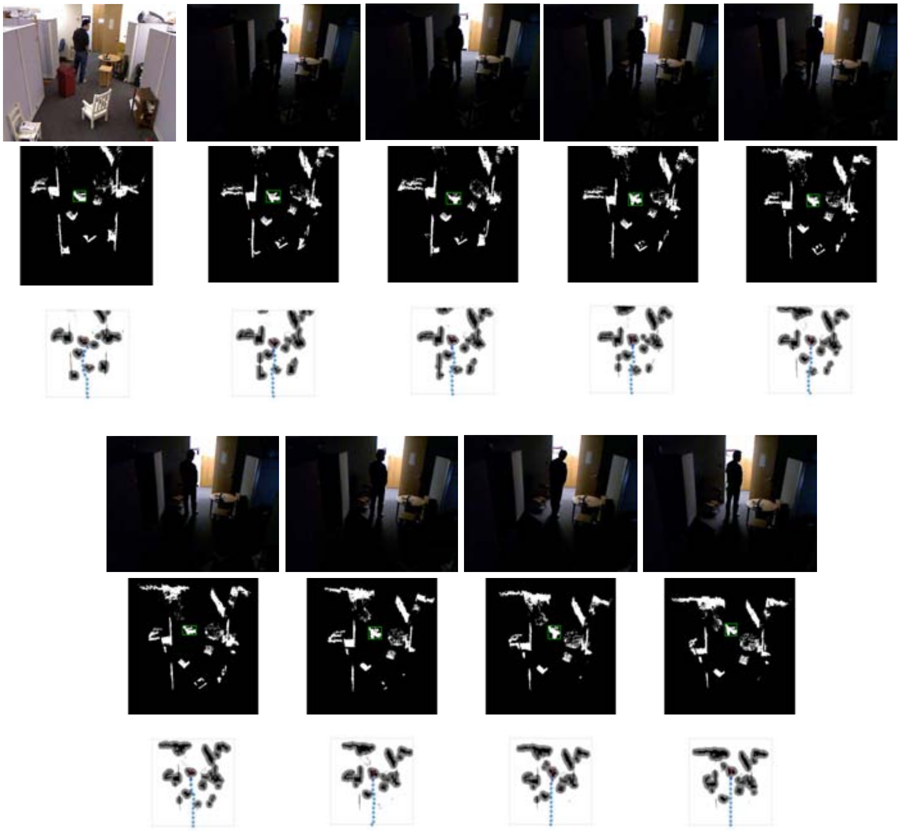


Fig. 13. Navigation results on dark indoor environments. The first row shows the side-view RGB images. The second row shows the GPP images and the green box indicating the tracked target. The third row shows the path planning results on GPP (color online).

8. Conclusion

In this paper, we focus on the indoor environment and utilize the depth information from RGB-D cameras in a novel way. The key contributions of the proposed method are: (1) a novel tracking representation based on GPP, (2) the detection process is performed in GPP by simple footprints segmentation, which is a category free and unsupervised detection method, (3) based on the detection results in GPP, a novel motion analysis named PLA is proposed on GPP, (4) PLA is based on optical flow in GPP. As clearly demonstrated, the optical flow in GPP is significantly more robust and stable than the commonly used optical flow in RGB images.

By back projecting the detection and tracking results to the original RGB images, we obtain a system for object tracking in RGB images. The key property of the proposed method is that no computation is actually performed in the side-view RGB images.

As our experimental results demonstrate, the proposed approach outperforms the state-of-the-art appearance-based object detection and tracking algorithms by a few orders of magnitude. We also achieve excellent results in robot navigation by tracking.

Acknowledgments

We would like to thank Zygmunt Pizlo and Yunfeng Li from Purdue University for their inspiring discussions. This work was supported by National Science Foundation under Grants OIA-1027897 and IIS-1302164, and by the National Natural Science Foundation of China (NSFC) Grants 61222308, 60903096 and 61173120, Fundamental Research Funds for the Central Universities HUST 2013TS115, the Program for New Century Excellent Talents in University.

References

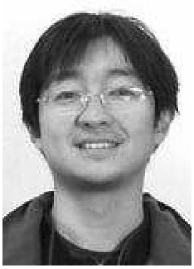
1. A. Adam, E. Rivlin and I. Shimshoni, Robust fragment-based tracking using the integral histogram, in *Proc. IEEE Computer Vision and Pattern Recognition (NY, USA, CVPR)* (2006), pp. 798–805.
2. M. Arie, A. Moro, Y. Hoshikawa, T. Ubukata, K. Terabayashi and K. Umeda, Fast and stable human detection using multiple classifiers based on subtraction stereo with HOG features, *IEEE Int. Conf. Robotics and Automation (Shanghai, China, ICRA)* (2011), pp. 868–873.
3. S. Avidan, Support vector tracking, *IEEE Trans. Pattern Anal. Mach. Intell.* **29** (2007) 261–271.
4. B. Babenko, M. H. Yang and S. Belongie, Robust object tracking with online multiple instance learning, *IEEE Trans. Pattern Anal. Mach. Intell.* **27** (2004) 1619–1632.
5. S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black and R. Szeliski, A database and evaluation methodology for optical flow, *Int. J. Comput. Vis.* **92** (2011) 1–31.
6. S. Belongie, J. Malik and J. Puzicha, Object detection with discriminatively trained part-based models, *IEEE Trans. Pattern Anal. Mach. Intell.* **27** (2004) 1619–1632.

7. J. Berclaz, F. Fleuret, E. Turetken and P. Fua, Multiple object tracking using K-shortest paths optimization, *IEEE Trans. Pattern Anal. Mach. Intell.* **33** (2011) 745–770.
8. K. Bernardin and R. Stiefelhagen, Evaluating multiple object tracking performance: The CLEAR MOT metrics, *J. Image Video Process.* **2008** (2008) 1–10.
9. D. Burschka, S. Lee and G. Hager, Stereo-based obstacle avoidance in indoor environments with active sensor re-calibration, *IEEE Int. Conf. Robotics and Automation (Washington, DC, USA, ICRA)* (2002), pp. 2066–2072.
10. A. Collet, M. Martinez and S. S. Srinivasa, The moped framework: Object recognition and pose estimation for manipulation, *Int. J. Robot. Res.* **30** (2011) 1284–1306.
11. D. Comaniciu, V. R. Member and P. Meer, Kernel-based object tracking, *IEEE Trans. Pattern Anal. Mach. Intell.* **25** (2003) 564–575.
12. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection, in *Proc. IEEE Computer Vision and Pattern Recognition (San Diego, CA, USA, CVPR)* (2005), pp. 886–893.
13. C. Desai, D. Ramanan and C. Fowlkes, Discriminative models for multi-class object layout, in *Proc. IEEE Computer Vision and Pattern Recognition (Colorado, USA, CVPR)* (2011), pp. 229–236.
14. A. Ess, B. Leibe, K. Schindler and L. van Gool, A mobile vision system for robust multi-person tracking, *IEEE Computer Society Conf. Computer Vision and Pattern Recognition (Anchorage, AK, USA, CVPR)* (2008), pp. 1–8.
15. A. Ess, B. Leibe, K. Schindler and L. Van Gool, Moving obstacle detection in highly dynamic scenes, *IEEE Int. Conf. Robotics and Automation (Kobe, Japan, ICRA)* (2009), pp. 56–63.
16. A. Ess, B. Leibe and L. Van Gool, Depth and appearance for mobile scene analysis, *IEEE Interestial Conf. Computer Vision (Rio de Janeiro, Brazil, ICCV)* (2007), pp. 1–8.
17. J. Fan, X. Shen and Y. Wu, "Scribble tracker: A matting-based approach for robust tracking", *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(8) (2012) 1633–1644.
18. J. Fan, Y. Wu and S. Dai, Discriminative spatial attention for robust tracking, in *Proc. European Conf. Computer Vision (Heraklion, Crete, Greece, ECCV)* (2010), pp. 480–493.
19. P. F. Felzenszwalb, R. B. Girshick, D. McAllester and D. Ramanan, Object detection with discriminatively trained part-based models, *IEEE Trans. Pattern Anal. Mach. Intell.* **27** (2004) 1619–1632.
20. M. Fischler and R. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *ACM Commun.* **24** (1981) 381–395.
21. D. M. Gavrila and S. Munder, Multi-cue pedestrian detection and tracking from a moving vehicle, *Int. J. Comput. Vis.* **32** (2010) 1627–1645.
22. H. Gong, J. Sim, M. Likhachev and J. Shi, Multi-hypothesis motion planning for visual object tracking, *IEEE Interestial Conf. Computer Vision (Barcelona, Spain, ICCV)* (2011), pp. 619–626.
23. H. Grabner, M. Grabner and H. Bischof, Real-time tracking via on-line boosting, *British Machine Vision Conf. (Edinburgh, UK, BMVC)* Vol. 1 (2006).
24. P. Hart, N. Nilsson and B. Raphael, A formal basis for the heuristic determination of minimum cost paths, *IEEE Trans. Syst. Sci. Cybern.* **4** (1968) 100–107.
25. M. Harville and D. Li, Fast, integrated person tracking and activity recognition with plan-view templates from a single Stereo Camera, *IEEE Computer Society Conf. Computer Vision and Pattern Recognition (Washington, DC, USA, CVPR)* (2004), pp. 398–405.
26. http://www.wanyrobotics.com/store/.

27. http://www.xbox.com/en-US/Kinect.
28. http://www.ptgrey.com/products/bumblebee2/.
29. J. Kwon and K. M. Lee, Visual tracking decomposition, in *Proc. IEEE Computer Vision and Pattern Recognition (San Francisco, CA, USA, CVPR)* (2010), pp. 1269–1276.
30. J. Kwon and K. M. Lee, Tracking by sampling trackers, *IEEE Interestial Conf. Computer Vision (Barcelona, Spain, ICCV)* (2011), pp. 1195–1202.
31. Y. Li, T. Sawada, L. J. Latecki, R. Steinman and Z. Pizlo, A tutorial explaining a machine vision model that emulates human performance when it recovers natural 3D scenes from 2D images, *J. Math. Psyc.* **56** (2012) 217–231.
32. C. Liu, Beyond pixels: Exploring new representations and applications for motion analysis, Doctoral Thesis, MIT (2009).
33. D. Lowe, Distinctive image features from scale-invariant keypoints, *Int. J. Comput. Vis.* **60** (2004) 91–110.
34. J. Ma, T. H. Chung and J. Burdick, A probabilistic framework for object search with 6-DOF pose estimation, *Int. J. Robot. Res.* **30** (2011) 1284–1306.
35. X. Mei and H. Ling, Robust visual tracking and vehicle classification via sparse representation, *IEEE Trans. Pattern Anal. Mach. Intell.* **33** (2011) 2259–2272.
36. H. Pirsiavash, D. Ramanan and C. Fowlkes, Globally-optimal greedy algorithms for tracking a variable number of objects, *IEEE Computer Society Conf. Computer Vision and Pattern Recognition (Colorado, USA, CVPR)* (2011), pp. 1201–1208.
37. D. Ross, J. Kim, R.-S. Lin and M.-H. Yang, Incremental learning for robust visual tracking, *Int. J. Comput. Vis.* **77** (2008) 125–141.
38. S. Thrun, W. Burgard and D. Fox, *Probabilistic Robotics*, Intelligent Robotics and Autonomous Agents (The MIT Press, 2005).
39. B. Yang, C. Huang and R. Nevatia, Learning affinities and dependencies for multi-target tracking using a CRF model, *IEEE Computer Society Conf. Computer Vision and Pattern Recognition (Colorado, USA, CVPR)* (2011), pp. 1233–1240.
40. X. Yang, H. Liu and L. J. Latecki, Contour-based object detection as dominant set computation, *Pattern Recogn.* **45** (2012) 1927–1935.
41. B. Yang and R. Nevatia, An online learned CRF model for multi-target tracking, *IEEE Computer Society Conf. Computer Vision and Pattern Recognition (Providence, RI, USA, CVPR)* (2012), pp. 2034–2041.
42. B. Yang and R. Nevatia, Multi-target tracking by online learning of non-linear motion patterns and robust appearance models, *IEEE Computer Society Conf. Computer Vision and Pattern Recognition (Providence, RI, USA, CVPR)* (2012), pp. 1918–1925.
43. B. Yang and R. Nevatia, Online learned discriminative part-based appearance models for multi-human tracking, in *Proc. European Conf. Computer Vision (Florence, Italy, ECCV)* (2012), pp. 484–498.
44. Z. Yao, Y. Zhou, J. Liu and W. Liu, A fast and effective appearance model-based particle filtering object tracking algorithm, *Int. Conf. Pattern Recognition (Tsukuba, Japan, ICPR)* (2012), pp. 1475–1478.
45. D. Young and J. Ferryman, Pets metrics: On-line performance evaluation service, *Joint IEEE Int. Work-Shop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (Mumbai, India, VS-PETS)* (2005), pp. 317–324.
46. Y. Zhou, X. Bai, W. Liu and L. J. Latecki, Fusion with diffusion for robust visual tracking, *Neural Information Processing Systems Conf. (Lake Tahoe, USA, NIPS)* (2012), pp. 2987–2995.
47. Y. Zhou, C. Rao, Q. Lu, X. Bai and W. Liu, Multiple feature fusion for object tracking, *Intelligent Science and Intelligent Data Engineering* (Xian, China, 2012), pp. 145–152.

48. Y. Zhou, J. Wang, Q. Zhou, X. Bai and W. Liu, Shape matching using points co-occurrence pattern, *IEEE Int. Conf. Image and Graphics (Hefei, Anhui, China, ICIG)* (2011), pp. 344–349.

Yu Zhou received his B.S. degree in Electrical Engineering from Wuhan Polytechnic University (WHPU), Wuhan, P.R. China in 2007, and his M.S. degree in Electronics and Information Engineering from Huazhong University of Science and Technology (HUST), Wuhan, P.R. China in 2009. He is currently working towards his Ph.D. at HUST. From January 2012 to January 2013, he worked in the Department of Computer Sciences and Information, Temple University. His research interests include computer vision and pattern recognition.



Yinfei Yang received his B.E. degree in Computer Science from Nanjing University of Posts and Telecommunications in 2009. He received his M.S. degrees in Computer Science from the Saint Joseph's University and University of Pennsylvania in 2012 and 2013. He was a research assistant at Grasp Lab of University of Pennsylvania from 2012 to 2013. He joined Amazon at 2013, where he is currently a software developing engineer of corporate applications.

Meng Yi received her B.S. degree in Computer Science from Beihang University (BUAA), Beijing, P.R. China in 2006. Currently she is working toward her Ph.D. at Temple University, Philadelphia, USA. Her research interests include computer vision and pattern recognition.

Xiang Bai received his B.S. and M.S. degrees both in Electronics and Information Engineering from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2003 and in 2005, respectively. He obtained his Ph.D. from HUST in 2010. From January 2006 to May 2007, he worked in the Department of Computer Sciences and Information, Temple University. From October 2007 to October 2008, he worked at the University of California, Los Angeles as a joint PhD student. Currently he is a faculty member of EI Department, HUST. His research interests include computer graphics, computer vision, and pattern recognition.

Wenyu Liu received his B.S. degree in Computer Science from Tsinghua University, Beijing, China, in 1986, and his M.S. degree and Ph.D. in Electronics and Information Engineering from Huazhong University of Science and Technology (HUST), Wuhan, China, in 1991 and 2001, respectively. He is currently a Professor and Associate Dean of the Department of Electronics and Information Engineering, HUST. His current research interests include computer graphics, multimedia information processing, and computer vision. He is a member of the IEEE and the IEEE Systems, Man, and Cybernetics Society.

Longin Jan Latecki received his Ph.D. in Computer Science from Hamburg University, Germany, in 1992. He is a Professor of Computer Science at Temple University, Philadelphia. His main research interests include shape representation and similarity, object detection and recognition in images, robot perception, machine learning, and digital geometry. He has published 200 research papers and books. He is an editorial board member of *Pattern Recognition and International Journal of Mathematical Imaging*. He received the annual Pattern Recognition Society Award together with Azriel Rosenfeld for the best article published in the journal *Pattern Recognition* in 1998. He is the recipient of the 2000 Olympus Prize, the main annual award, from the German Society for Pattern Recognition (DAGM).