Multi-hypothesis Motion Planning for Visual Object Tracking

Haifeng Gong[†], Jack Sim[†], Maxim Likhachev[‡], Jianbo Shi[†]
[†] GRASP Lab, University of Pennsylvania
[‡] Robotics Institute, Carnegie Mellon University
hfgong@seas.upenn.edu, {jiwoong, jshi}@cis.upenn.edu, maxim@cs.cmu.edu

Abstract

In this paper, we propose a long-term motion model for visual object tracking. In crowded street scenes, persistent occlusions are a frequent challenge for tracking algorithm and a robust, long-term motion model could help in these situations. Motivated by progresses in robot motion planning, we propose to construct a set of 'plausible' plans for each person, which are composed of multiple long-term motion prediction hypotheses that do not include redundancies, unnecessary loops or collisions with other objects. Constructing plausible plan is the key step in utilizing motion planning in object tracking, which has not been fully investigate in robot motion planning. We propose a novel method of efficiently constructing disjoint plans in different homotopy classes, based on winding numbers and winding angles of planned paths around all obstacles. As the goals can be specified by winding numbers and winding angles, we can avoid redundant plans in the same homotopy class and multiple whirls or loops around a single obstacle.

We test our algorithm on a challenging, real-world dataset, and compare our algorithm with Linear Trajectory Avoidance and a simplified linear planning model. We find that our algorithm outperforms both algorithms in most sequences.

1. Introduction

In crowded street scenes, frequent occlusions, combined with appearance changes, lead to ambiguous data association or 'drifting' in tracking. Many of these occlusions could be dealt with using a long-term motion model. Motivated by progresses in robot motion planning, we propose to construct a set of 'plausible' plans for each person, which are composed of multi-hypotheses for motion prediction without redundancies, unnecessary loops or collisions with other objects. Constructing 'plausible' plans is the key property desired by visual object tracking but has not been fully investigated in robot motion planning, which focuses on figuring out the most efficient path. We introduce a novel method of efficiently constructing disjoint plans in different homotopy classes, based on winding numbers and winding angles of planned paths around all obstacles. As the goals can be specified by winding numbers and winding angles, we avoid redundant plans in the same homotopy class or multiple whirls and loops around a single obstacle, even with obstacles of varying different sizes and shapes.

There are two key factors distinguishing our motion model from traditional motion models. First, we explicitly model pedestrian trajectories as goal-directed obstacle-avoiding paths using multiple hypotheses. Second, we create more flexible and realistic hypotheses for possible pedestrian trajectories than others; simpler models of dynamic social behavior [10] are limited in expressive power because they use single hypotheses and short-term predictions.

For each person, our planner maintains multiple hypotheses for future paths as they move in the environment, creating a 'virtual simulation' of intended pedestrian motion. When a person is visible, we track them, and use their trajectory to narrow down the set of plausible goals/planned paths. When a person becomes occluded, we create multiple hypotheses that predict their re-appearance based on the plausible set of goals/planned paths provided by the planner. Figure 1 illustrates this process.

We apply our motion model on batch-mode tracklets association, where we model, in the tracklet matching cost, agreement between goal-oriented motion plans. We test our method on data collected from a car mounted with a stereo camera pair driven across an urban city.

2. Related Work

When multiple objects have a similar appearance, or when occlusion happens and appearance features are corrupted, better motion model can improve tracking. Recently, object-interaction based motion models have attracted much attention. The most similar approaches to our method are those of [6] and [9, 10]. Helbing et al[6] introduced a dynamic social behavior model for simulating people behavior. They model the velocities and accelerations of people in crowds using three terms: 1) a term describing

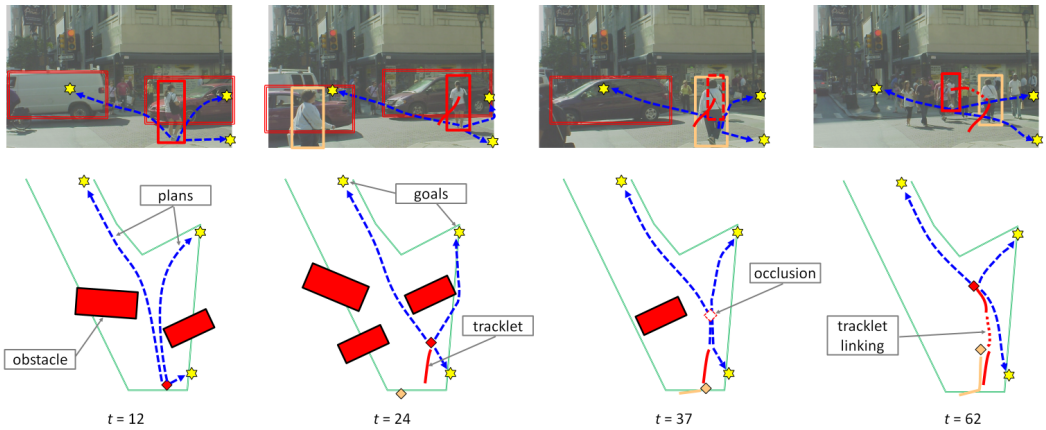

Figure 1. Tracking by planning. We endow each tracked person with a planning agent for tracking under occlusion and ambiguity. The person in the bold red bounding box, at $t = 12$, crosses the road during a red-light. She zig-zags to avoid coming cars, and is occluded by the person with similar appearance at $t = 37$. By estimating obstacle avoiding path, we succeed in tracking her.

the acceleration towards the desired velocity of motion; 2) a term reflecting that a pedestrian keeps a certain distance from other pedestrians and borders; and 3) a term modeling attractive effects of groups of people. Luber et al[9] introduced the social force model proposed by Helbing et al into visual object tracking by combining it with a Kalman filter, resulting in a more realistic prediction model. Pellegrini et al[10] proposed a goal-directed short-term obstacle avoidance model, called Linear Trajectory Avoidance (LTA). They used an energy function consist of two terms: the first encourages a large distance from obstacles over time, and the second encourages walking toward a manually specified goal with a given speed. Our method differs from the above mentioned ones in two aspects, 1) we give multiple hypotheses explicitly, which is more likely to cover all possible trajectories over long occlusions; 2) we predict entire paths rather than just short-term velocities.

Multiple people and multiple hypothesis tracking is an active research area since the work of [11]. For example, [3] prunes an exponentially growing tree of Kalman filters by determining the k-best hypotheses in polynomial time; [7] relaxes the association in object tracking as a multi-path search problem. Our work is different from these methods in that we discard the Markov assumption that is adopted in the traditional literature.

Though there also have been large body of work in motion planning in robotics [8], for visual object tracking, traditional robotics planning is insufficient. Most planning algorithms find the optimal solution for reaching a goal with minimal cost. However, it is difficult to model such a cost for each person in a complex real life setting. Instead, it is more important to construct multi-hypothesis planning to cover all 'plausible' plans. Research on path planning with multiple hypotheses has been paid less attention. In robotics multi-hypothesis planning has been explored under the name of 'homotopy constrained planning' [5][2]. A homotopy class of trajectories [2] is defined by the set of trajectories joining same start and end points which can be smoothly deformed into one another without intersecting obstacles. Homotopy class-constrained planning finds the best solution in certain allowed homotopy classes. That is, although they split the solution space into homotopy classes, they still focus on finding best solutions in one or more given homotopy classes for a robot to follow. This is a little different from our problem. We want a set of paths to best cover as many as possible object trajectories. Our problem involves enumerating the most likely homotopy classes efficiently as well as finding the best solution in the specified homotopy classes. Although the latter has been solved in [2], we propose a more efficient solution. The former remains unexplored, and we solve it by indexing the homotopy classes with winding numbers around obstacles.

3. Multi-hypothesis Planning

3.1. Homotopy Class of Planning by L-value

For multi-hypothesis planning, we want to find multiple non-redundant paths. To be precise, if two paths joining same start and end points can be smoothly deformed into one another without intersecting obstacles, they are redundant (see Figure 2). This set of redundant paths is called a homotopy class [2]. In [2], the authors studied the problem of finding least-cost paths restricted to, or excluded from, a specific homotopy class. They represent the environment of the robot as a complex plane and make use of the Cauchy Integral Theorem to define a complex index for each homotopy class.

In this subsection, we give a brief summary of the method of [2], on which our algorithm is based. Let z be a point in the complex plane, z_b be the start point and z_g be the goal. A path $\gamma(s)$ is a complex function of arc length parameter $s \in [0, T]$, with constraints $\gamma(0) = z_b$ and $\gamma(T) = z_g$.

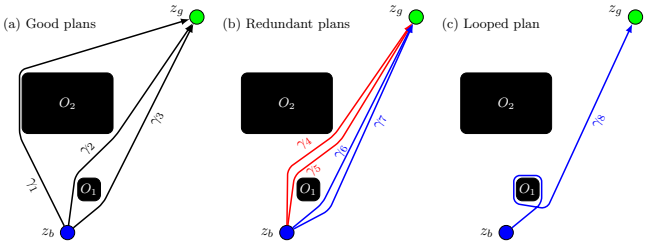

Figure 2. Examples of plausible plans and bad plans. O_1 and O_2 are two obstacles. γ_i are possible paths. z_b and z_g are the start point and goal respectively. (a) A set of good plans in different homotopy classes that have no unnecessary loops. (b) Two groups of redundant plans: γ_4 and γ_5 belong to the same homotopy class, as do γ_6 and γ_7. (c) A path makes an unnecessary loop around obstacle O_1. [2] can efficiently avoid bad plans like (b) by using homotopy classes, but it cannot avoid the bad plans like (c) because it uses a simple complex number to index the homotopy classes, which contains no information about loops. [2] enumerates all plans in all homotopy classes in the order of path costs. Assume that plain arc length is used as the cost, γ_3 is the first to be found, then γ_2. Because γ_8 is shorter than γ_1, it is found next. [2] will make a couple of loops before reaching the next plausible plan γ_1.

To distinguish different homotopy classes, a complex *obstacle marker function* is defined as

$$F(z) = \frac{f_0(z)}{(z-\zeta_1)(z-\zeta_2)\cdots(z-\zeta_N)} \tag{1}$$

where $f_0(z)$ is a complex Holomorphic function and ζ_i is a point in the area covered by obstacle i in the complex plane. Using the Cauchy Integral Theorem, they showed that two trajectories $\gamma_1(s)$ and $\gamma_2(s)$ connecting the same pair of points lie in the same holotopy class if and only if

$$\int_{\gamma_1} F(z)dz = \int_{\gamma_2} F(z)dz \tag{2}$$

given the assumption that $f_0(z)$ meets certain conditions. Therefore they use the L-value, defined as

$$L(\gamma) = \int_{\gamma} F(z)dz \tag{3}$$

to index homotopy classes. Note that although the L-value is a continuous complex number, it only has discrete number of possible values, given start point and goal.

Starting from a standard graph based planning configuration, each vertex is augmented with multiple L-values, and becomes multiple vertices to form an augmented graph. Goals of different homotopy classes are different nodes in this graph. The edges of the original graph are augmented similarly with the increments of L-values. As such, standard graph search algorithm can be used to find the shortest path from the start point to the goal in a specified homotopy class.

However, one cannot directly apply [2] for person tracking:

1. When obstacles differ greatly in size, [2] performs poorly in enumerating all the homotopy classes of plans. Their algorithm will focus on finding shortest paths with multiple loops around small obstacles before finding a path around the other side of a larger obstacle. This occurs because although their L-value can distinguish one homotopy class from other ones, it cannot carry other necessary information such as how many loops a homotopy class contains. The authors suggest discarding smaller obstacles to overcome this problem. This suggestion is not acceptable in street scene tracking, where people and cars are dynamic obstacles and have different sizes. We cannot discard all people and consider only cars as obstacles. Figure 2 shows details about this point.
2. Obstacle marker function (1) must be carefully chosen for numeric stability of L-values in real-world applications.
3. Their representation of state space is an infinite augmented graph. This occurs because that the L-value does not record number of loops around the obstacle, and so has to allow infinite number of them. However, in real-world visual tracking, it is better to keep the search space finite.

We propose replacing L-value with a more informative index, that incorporates the number of loops around obstacles. This allow us to screen out any paths with many loops, which are unlikely to be the paths that people actually take.

3.2. From L-value to winding numbers

Following [2], we use the complex plane to describe the configuration space, i.e., ground and obstacles. Let us consider the L-value of a plan γ with respect to a single obstacle,

$$L = \int_{\gamma} \frac{f(z)}{z - z_0} dz \tag{4}$$

where z_0 is a point on the obstacle and $f(z)$ can be any complex holomorphic function such that $f(z_0) \neq 0$. L-values for a single obstacle must be in the discrete set of

$$\{k * 2\pi i f(z_0) + L_0 : k \in \mathbb{Z}\}, \tag{5}$$

where L_0 is the L-value of the path from start point to goal at right side with no loop. Thus we can use k to distinguish homotopy classes with respect to one obstacle which we call winding number. For a plausible path, the values of k will likely be 0 or -1, meaning 'go-right' or 'go-left' around the obstacle. When $k > 0$, it indicates a path to the right of the obstacle that includes k loops around it. Similarly $k < -1$ indicates a path to the left of the obstacle that includes $-k-1$ loops around it. In most cases, a plausible path will have $k \in \{-1, 0\}$. Though for an obstacle or environment with irregular shape, the plausible path may has $k < -1$ or $k > 0$, in street scenes, we can hardly meet this situation. Therefore, we only consider $k \in \{-1, 0\}$ as plausible in implementation.

By letting k_i be the k-value associated with the i-th obstacle, we can denote a homotopy class with respect to all

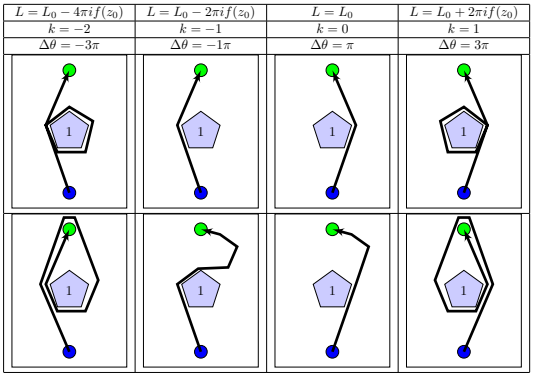

Figure 3. Winding numbers and winding angles for one obstacle. First row, L-values. Second row, k-values. Third row, winding angles. Fourth row, example plans. Fifth row, more example plans. One can see that $k > 0$ or $k < -1$ indicates that there are loops around obstacles.

obstacles as an integer vector (vector of winding numbers, or k-**vector**)

$$\mathbf{k} = (k_1, k_2, \cdots, k_N)^T. \tag{6}$$

Theorem 1. *Two trajectories γ_1 and γ_2 with k-vectors $\mathbf{k}_1$ and $\mathbf{k}_2$ connecting the same points lie in the same homotopy class if and only if $\mathbf{k}_1 = \mathbf{k}_2$.*

Proof. if-clause: If $\mathbf{k}_1 = \mathbf{k}_2$, then L-values for all obstacles are same, which means no obstacle is enclosed by the closed contour formed by γ_1 and γ_2, following Theorem 1 in [2]. Therefore, they lie in the same homotopy class. *only-if*-clause: If they lie in the same homotopy class, then they enclose no obstacle, and therefore have same value in each entry of their k-vectors. □

Given a start point, a goal, and a set of obstacles, a one-to-one map can be established between the set of all homotopy classes and the set of vectors of winding numbers. The theorem above states that vectors of winding numbers give a complete description of the topology of feasible trajectories given an environment, a starting location, and goal.

3.3. From winding numbers to winding angles

In Eq. (4), we simply choose $f(z) = 1$ to be a constant. Given a path γ, if we write it in parametric form,

$$\gamma(s) = z_0 + r(s)\exp[i\theta(s)] \tag{7}$$

where $s \in [0, T]$ is arc length parameter, Eq. (4) can be computed in closed form as

$$L = \log r(T) - \log r(0) + i[\theta(T) - \theta(0)], \tag{8}$$

where the real part is constant for all possible paths γ with given start point and goal since $r(0) = \|z_0 - z_b\|$ and $r(T) = \|z_0 - z_g\|$. The imaginary part

$$\Delta\theta = \theta(T) - \theta(0) = \Delta\theta_0 + 2k\pi \tag{9}$$

may differ by $2k\pi$, where k is also a winding number. We call $\Delta\theta$ the winding angle of γ with respect to obstacle z_0. Each path has a vector of winding angles with respect to all obstacles, $\Delta\Theta = (\Delta\theta_1, \Delta\theta_2, \cdots, \Delta\theta_N)^T$. Now, we can build our algorithm using winding angles directly, and do not need to consider L any more. See Figure 3 for examples of winding numbers and winding angles.

3.4. Augmented Graph

Like [2], we use a graph based search algorithm. We begin with neighborhood graph G, in which each grid point on ground not occupied by an obstacle is a vertex, and each pair of neighboring points are connected by an edge. Edge weights are the costs of moving from one vertex to another. If we simply use path length as the cost, the shortest path on this graph is the shortest path in the configuration space subject to no collisions. Each vertex in G is represented by its coordinate on ground z.

We augment this graph with winding angle to create an augmented graph $\bar{G}$. That is, we equip both vertices with winding angles and edges with increments of winding angles. We can choose a set of possible vectors of winding numbers $K = \{\mathbf{k}_i : i = 1, \cdots, |K|\}$, in which $|K|$ is the number of elements in K. If we choose 3 obstacles, we have $|K| = 2^3 = 8$, because for each obstacles, we have 2 choices of $k \in \{-1, 0\}$. Then, for each vertex z in G, we have a set of vectors of winding angles $A_z = \{\Delta\Theta_z^i : i = 1, \cdots, |K|\}$, each of which corresponds to a winding number vector in K through Eq. (9). A vertex of the augmented graph $\bar{G}$ is represented by $(z, \Delta\Theta_z^i)$, that is, the pair of each z and each of its winding angle vector $\Delta\Theta_z^i$. If the number of vertices of G is $|G|$, then the number of vertices in $\bar{G}$ is $|K| \times |G|$. Let e be the edge connecting two vertices z and z' in the original graph G. The edge has a fixed winding angle vector, $\Delta\Theta_e$. For the vertices $(z, \Delta\Theta_z^i)$ and $(z', \Delta\Theta_{z'}^{i'})$, we connect them if $\Delta\Theta_{z'}^{i'} = \Delta\Theta_z^i + \Delta\Theta_e$. In the augmented graph, a goal is split into multiple vertices according to winding angle vector. See Figure 4 for more explanation.

The graph weights are defined in the following way. Given an environment with static and dynamic obstacles, we compute distance transformations of both a static obstacle map and a dynamic obstacle map. Let D_{st} and D_{dyn} be the two distance transformations. We define a vertex weight map as $W(z) = \alpha_0 + \alpha_1 D_{\text{st}}(z) + \alpha_2 D_{\text{dyn}}(z)$, where α_j are weights that provide a trade-off between the three terms. The weight of an edge e connecting z and z' is defined as the average of the weights of the vertices it connects, multiplied by the distance between them, $W(e) = \frac{1}{2}\{W(z) + W(z')\}\|z - z'\|$.

We use Dijkstra's algorithm to search the augmented graph. If too many obstacles are present, we select key obstacles by first finding the shortest path in the original

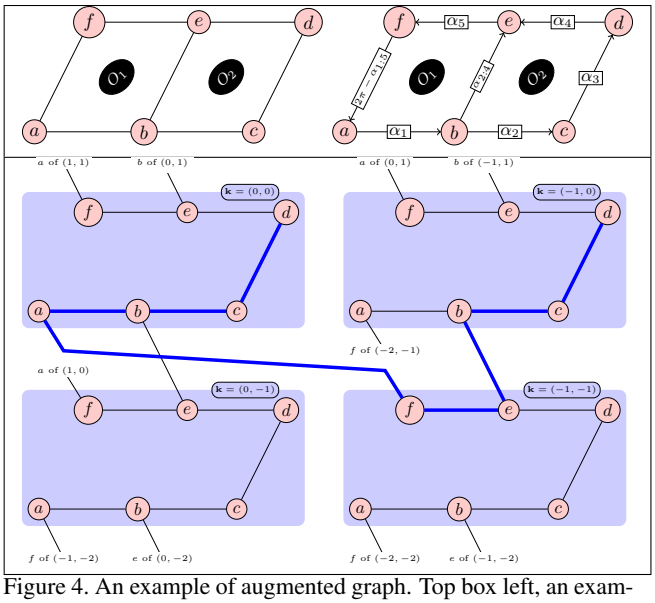

Figure 4. An example of augmented graph. Top box left, an example graph, with 6 nodes and 2 obstacles. a is the start point and d is the goal. Top box right, the winding angles on edges. Bottom box, the augmented graph, with four $\mathbf{k}$ vectors. Each $\mathbf{k}$ correspond to a layer with 6 nodes marked by a shaded panel, that is, each node is split into 4 in the augmented graph. From a in the first layer, there is a shortest path to d in each of the layers. The shortest paths from a in $\mathbf{k} = (0, 0)$ to d in $\mathbf{k} = (0, 0)$ and $\mathbf{k} = (-1, 0)$ are shown in bold blue.

neighborhood graph G, then keeping the obstacles close to the shortest path as key obstacles. Only the key obstacles are considered for computing winding numbers.

4. Tracking by Planning

We test our motion model in a batchmode tracking by detection framework. During tracking, when a person is partially or fully occluded, we estimate his position by planning. Tracking a person in the visible state leads to a short trajectory that we call a tracklet. A conservative threshold is used to terminate the trajectory when the tracking score becomes too low. After termination, the same person may be picked up again by the detection algorithm, and tracked to produce associated tracklets. After tracklets are obtained, we can link them using both appearance and planning consistency.

4.1. Criteria for tracklets linking by planning

We defer the discussion of person detection and initial tracking until Section 4.3. For now, we assume that we have a set of tracklets $\mathcal{T} = \{F_1, \cdots, F_{N_{\text{Tr}}}\}$, where F_i is the i-th tracklet, and N_{Tr} is the total number of tracklets. Each tracklet is described by $F_i = (t_0^i, t_1^i, \mathbf{x}_{t_0^i}^i, \cdots, \mathbf{x}_{t_1^i}^i)$, where t_0^i is the start time of F_i, t_1^i is the end time of F_i and $\mathbf{x}_t^i$ is the object position at time t. Note that $\mathbf{x}_t^i$ is defined in a fixed 3D world coordinate system defined by the initial camera. We first measure the people's positions in the stereo image frame, and map it to a fixed 3D world frame using the egomotion estimation of the camera.

We then link and extend these tracklets, $\mathcal{T}$, into complete trajectories, using the 'estimated' partial/full occlusion position to explain away the 'gap' formed by the tracklets. Let $L_{i,j}$ be the indicator of linking i-th and j-th tracklet:

$$L_{i,j} = \begin{cases} 1 & F_i \to F_j \\ 0 & \text{otherwise} \end{cases} . \tag{10}$$

To link tracklets into plausible goal-directed obstacle-avoiding paths, we design the following criterion for tracking:

$$\max_L \epsilon(L) = \sum_{i,j:L_{i,j}=1} [S_{\text{App}}(i,j) + \alpha S_{\text{Plan}}(i,j)] - \beta|L| \tag{11}$$

where $S_{\text{App}}(i,j)$ measures appearance similarity between tracklets F_i and F_j, $S_{\text{Plan}}(i,j)$ measures 1) how consistent F_i and F_j are with a plausible goal directed path; and 2) how partial occlusion in the gap can be explained by appearance of F_i and F_j. We introduce α to trade off the two scores and $\beta \neq 0$ to prevent aggressive linking. The criterion (11) is subject to the following constraints: $L_{i,j} \in \{0,1\}$, $\sum_i L_{i,j} \leqslant 1$, $\sum_j L_{i,j} \leqslant 1$, and $L_{i,j} = 0, \forall (i,j) \in \text{InvalidSet}$, where InvalidSet is used to exclude those links that indicates impossibly large speeds, too long gaps or time back-tracking. We seek an approximate solution using Linear Programming.

4.2. Planning score

The planning score is given by finding the best planned path to fill the gap between tracklet i and j. The best path is compatible with tracklet i and tracklet j geometrically, and allows possible partial matches by appearance during occlusions. We use the following score: $S_{\text{Plan}}(i,j) = \max_{r \in \text{paths}} -\text{Dist}(r, F_i) - \text{Dist}(r, F_j) + S_{\text{Occl}}(F_i, F_j, r)$, where $\text{Dist}(r, F_i)$ is the distance between path r and tracklet F_i and $S_{\text{Occl}}(F_i, F_j, r)$ is the score for picking up the partial occlusions along the gap. To reduce computation, we prune paths whose costs are higher than the minimal one above a threshold.

We compute the distance between a tracklet and planned path as follows. First, we shorten the tracklet by keeping only the last M frames of F_i and first M frames of F_j, giving the shortened tracklets F_i' and F_j'. Let $x_1, \cdots, x_M$ be the tracked positions in F_i' or F_j', and let l be the arc length of the shortened tracklet. Let p be the point on the path r which is nearest to the start point of F_i' or F_j'. Using p as start point, we can obtain an arc on the path r with length l, which results in a shortened path r'. Finally, we divide r' into $M-1$ segments uniformly, to obtain M end

points: $r'_1, \cdots, r'_M$. We compute the distance $\text{Dist}(r, F_i) = \sum_m \|x_m - r'_m\|^2$.

We find possible partial matches (by appearance) during occlusions, to compute $S_{\text{Occl}}(F_i, F_j, r)$, which is defined by the score of two appearance models applied to the hallucinated trajectory bridging the gap. Given the path r, which does not connect F_i and F_j perfectly, we first compute the hallucinated trajectory connecting F_i and F_j by a diffusion equation and project it to both cameras to pick up possible partial matches during occlusions. The diffusion equation uses the ends of F_i and F_j as boundary conditions, and the differences of the adjacent points on the planned paths as guided gradients.

4.3. Appearance Feature

Adaptive appearance model. For a pedestrian, we divide his image patch into three parts: head, torso and legs. Using a part based representation allows us to reason under partial occlusion. For each part k at time t, we collect the color histogram using $8 \times 8 \times 8$ bins, denoted by $\mathbf{p}_t(k)$, and we also collect the histogram of surrounding background, denoted by $\mathbf{q}_t(k)$. We use simple color feature instead of more advanced shape features for simplicity and computation efficiency. The histograms are collected using subsampling. We maintain running means of the histograms as an object model: $\mathbf{f}_t = (1-\alpha) * \mathbf{f}_{t-1} + \alpha * \mathbf{p}_t$, $\mathbf{b}_t = (1-\alpha) * \mathbf{b}_{t-1} + \alpha * \mathbf{q}_t$. Denote $\text{Model}_t = (\mathbf{f}_t, \mathbf{b}_t)$ the object appearance model.

Tracklet creation. We use a detector based on [4] to detect peoples and cars in the current frame. To track a person in frame $t+1$ given the models of previous frame, Model_t, we measure two scores — *Consistent Score (S1)* to ensure that it is similar to foreground appearance model $\mathbf{f}_t$ and different from $\mathbf{b}_t$, and *Contrast Score (S2)* to ensure that the foreground is different from its surroundings in current frame. They are defined as follows; $\text{S1}(\text{Model}_t, \mathbf{p}_{t+1}) = \frac{1}{2}\sum_k \sum_{\text{bin}} \mathbf{p}_{t+1}(k) \log \frac{\mathbf{f}_t(k)}{\mathbf{b}_t(k)} + \frac{1}{2}\sum \mathbf{f}_t(k) \log \frac{\mathbf{p}_{t+1}(k)}{\mathbf{b}_{t+1}(k)}$, and $\text{S2}(\text{Model}_t, \mathbf{p}_{t+1}) = \sum_k \text{KL}\{\mathbf{p}_{t+1}(k) \| \mathbf{q}_{t+1}(k)\} = \sum_k \sum_{\text{bin}} \mathbf{p}_{t+1}(k) \log \frac{\mathbf{p}_{t+1}(k)}{\mathbf{q}_{t+1}(k)}$.

Appearance score. The appearance score in Eq. (11) is obtained by testing the appearance model of tracklet i on model of tracklet j and vice-versa.

5. Experiments

To test our algorithm we have collected a video from a moving vehicle in an urban city. The stereo images were collected at 1024×768 resolution and 6 FPS. We have a system for people detection (based on [4]), 3D scene layout/goal estimation, and camera ego-motion computation. The ground plane at the first frame of each sequences was calibrated and propagated over time using the ego-motion transformations. We estimated building planes and ground plane in each frame and intersected them to get street side lines. The goals are estimated by intersecting the street side lines, plus infinity points along the street. We only track people, but detect cars using [4]. When planning for a specified object, other objects are regarded as obstacles.

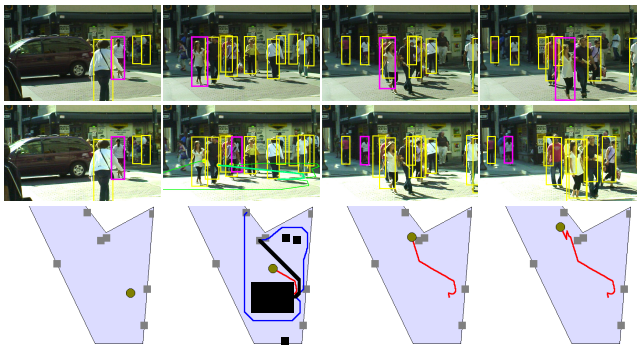

Figure 5. Top: tracking with linear linking. It drifts after occlusion. Middle: tracking with planning. We are able to pick up the entire trajectory of a pedestrian, despite the long occlusion. Bottom, top view of tracking with planning. The brown balls are current positions, the red curves are trajectories, the bold black curve is the selected plan, the blue curves are other plans, gray squares are possible goals. Black squares and rectangle are obstacles at planning time. Note that we plan in advance, therefore, the obstacles are other objects a few frames ago. Video frames are cropped for clarity. Better viewed in color.

We have picked 7 sequences which contains multiple people, and have interesting interactions and occlusions. Details of all sequences are shown in Table 1. There are total of 48 people in all the sequences. Many of these people cannot be tracked through the entirety of the sequence, because of the high occlusion rates. For comparison, we implement two baselines, 1) (LINEAR) tracklet linking without planning, that is, using a straight line as a plan to try to link the gaps, and 2) (LTA) Linear Trajectory Avoidance[10]. In the implementation of 1), we try straight line connection between all possible tracklet pairs, and pick the possible partial occlusions in the gaps. In the implementation of 2), we use LTA to predict a path from the end of one tracklet to the start of the other, and also pick up the possible partial occlusions in the gaps. We also compute the performace of our conservative tracklets (TRLET).

Table 2 shows the performance comparison in CLEAR Metrics[1]. We use 3 metrics: false alarm rate, miss rate and number of identity switches. The false alarms are caused by false alarms in detection and wrong occluded bounding boxes drifting to background. The misses are caused by misses in detection and failure to collect occluded bounding boxes. Note that we annotate the bounding boxes of an object even if it is totally occluded.

We divide the sequences into 3 difficulties, based on the number of occlusions. Seq #1, #2 and #3 are the most difficult ones. PLAN outperforms LINEAR and LTA in two of them and for the extremely difficult Seq #1, the results

of PLAN and LINEAR are similar. Seq #4 and #5 are of medium difficulty. PLAN outperforms LINEAR and LTA in #4 and LINEAR performs best in #5. Seq #6 and #7 are of relatively lower difficulty. Here, PLAN outperforms LINEAR and LTA in #6 and gets the same results as LINEAR in #7. One can see that the planning model helps deal with many occlusions, and is at least as good as linear prediction at low difficulties. The LTA model performs worst in all sequences because it cannot deal with long occlusions. TRLET always has the largest miss rate and almost the lowest false alarm rate[1]. Figure 5 and 6 demonstrate some of these results.

	# obj	# frames	# BB	#Occl. BB
seq #1	13	169	1139	471
seq #2	12	60	532	130
seq #3	7	35	210	125
seq #4	4	40	148	51
seq #5	5	112	211	46
seq #6	5	41	170	17
seq #7	2	27	54	16
Total	48	484	2464	856

Table 1. Test Videos with 3 difficulty levels according to the number of occluded bounding boxes. (BB = Bounding boxes.)

		miss rate	fa rate	id switch
seq #1	PLAN	**0.413**	0.089	9
	LINEAR	0.442	**0.070**	**8**
	LTA	0.488	0.214	8
	TRLET	0.511	0.089	17
seq #2	PLAN	**0.259**	**0.193**	**0**
	LINEAR	0.330	0.199	4
	LTA	0.366	0.310	6
	TRLET	0.407	0.112	6
seq #3	PLAN	**0.311**	0.223	**1**
	LINEAR	0.340	**0.200**	2
	LTA	0.476	0.445	6
	TRLET	0.580	0.043	15
seq #4	PLAN	**0.176**	**0.000**	**0**
	LINEAR	**0.176**	0.110	**0**
	LTA	0.270	0.212	**0**
	TRLET	0.412	0.074	**4**
seq #5	PLAN	0.137	0.032	**0**
	LINEAR	**0.123**	**0.016**	**0**
	LTA	0.189	0.090	0
	TRLET	0.193	0.023	0
seq #6	PLAN	**0.147**	0.194	**0**
	LINEAR	0.153	**0.152**	6
	LTA	0.211	0.394	5
	TRLET	0.276	0.054	20
seq #7	PLAN	**0.056**	**0.000**	**0**
	LINEAR	**0.056**	**0.000**	**0**
	LTA	0.203	0.157	**0**
	TRLET	0.241	0.000	**6**

Table 2. Performance evaluation in CLEAR metrics.

[1]Postprocessing in PLAN, LINEAR or LTA may furthur remove very short tracjectories. This postprocessing is not possible in TRLET because many of its results are very short.

6. Conclusion

We have presented a long-term multi-hypothesis motion model for visual object tracking based on motion planning. It was tested on tracking multiple people in a cluttered scene. Our planner solved the key problems of utilizing motion planning in visual object tracking, and the experiment results demonstrate the effectiveness of our motion model.

Acknowledgement

We would like to thank Weiyu Zhang, Ryan Kennedy, Katerina Fragkiadaki and Jeffrey Byrne for proof-reading. This work is supported by ONR MURI N00014-09-1-1052 and ARL RCTA W911NF-10-2-0016.

References

[1] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: The CLEAR MOT metrics. *EURASIP J. Image and Video Proc.*, 2008.

[2] S. Bhattacharya, V. Kumar, and M. Likhachev. Search-based path planning with homotopy class constraints. In *AAAI*, 2010.

[3] I. J. Cox and S. L. Hingorani. An efficient implementation and evaluation of Reid's multiple hypothesis tracking algorithm for visual tracking. In *ICPR*, 1994.

[4] P. Felzenszwalb, D. McAllester, and D. Ramaman. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[5] D. Grigoriev and A. Slissenko. Polytime algorithm for the shortest path in a homotopy class amidst semi-algebraic obstacles in the plane. In *Proc. of Int'l Symp. on Symbolic and Algebraic Computation*, 1998.

[6] D. Helbing and P. Molnár. Social force model for pedestrian dynamics. *Physical Review E*, 51(5):4282–4286, 1995.

[7] H. Jiang, S. Fels, and J. J. Little. A linear programming approach for multiple object tracking. In *CVPR*, 2009.

[8] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006.

[9] M. Luber, J. A. Stork, G. D. Tipaldi, and K. O. Arras. People tracking with human motion predictions from social forces. In *ICRA*, 2010.

[10] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *ICCV*, 2009.

[11] D. Reid. An algorithm for tracking multiple targets. *IEEE Trans. on Automatic Control*, 24(6):843–854, Dec 1979.

Figure 6. Image patches and bounding boxes over time. Each panel shows the bounding boxes of a pedestrian in two parts. The top parts show the image patches of ground truth (1st row), PLAN results (2nd row) and LINEAR results (3rd row). The number on each box is the frame number. They are trimmed on left or right for better visual effects. The bottom parts show video frames superimposed with bounding boxes. The magenta bounding boxes are current objects of interests. Yellow bounding boxes are other objects. The bold green lines are the planned routes that the objects follow. The thinner green lines are other planned paths (after pruning) that are not followed by the people. **Seq 1** shows subsampled patches from a 154-frame trajectory. A girl in black is first occluded by a pole for about 25 frames (0∼24), then occluded by a girl in red for 15 frames (38∼53) and finally occluded by a truck for 5 frames (124∼128). PLAN covers almost the whole trajectories, with some small drifts. LINEAR fails to link the two long occlusion. **Seq 2**, a woman is occluded for about 10 frames, PLAN catches up after occlusion, and picks up correct partial occlusions; but LINEAR drifts away to a detection false alarm. **Seq 3**, a man undergoes two short occlusions, is caught up by PLAN, but LINEAR and LTA terminate the trajectory too early. Video frames are cropped for clarity. Better viewed in color.