Joint Object Class Sequencing and Trajectory Triangulation (JOST)

Enliang Zheng, Ke Wang, Enrique Dunn, and Jan-Michael Frahm

The University of North Carolina, Chapel Hill

Abstract. We introduce the problem of joint object class sequencing and trajectory triangulation (JOST), which is defined as the reconstruction of the motion path of a class of dynamic objects through a scene from an unordered set of images. We leverage standard object detection techinques to identify object instances within a set of registered images. Each of these object detections defines a single 2D point with a corresponding viewing ray. The set of viewing rays attained from the aggregation of all detections belonging to a common object class is then used to estimate a motion path denoted as the object class trajectory. Our method jointly determines the topology of the objects motion path and reconstructs the 3D object points corresponding to our object detections. We pose the problem as an optimization over both the unknown 3D points and the topology of the path, which is approximated by a Generalized Minimum Spanning Tree (GMST) on a multipartite graph and then refined through a continuous optimization over the 3D object points. Experiments on synthetic and real datasets demonstrate the effectiveness of our method and the feasibility to solve a previously intractable problem.

1 Introduction

Reconstruction from photo collections has attracted significant attention in the last decade, enabling systems to build 3D models from entire city datasets of millions of images[14]. Despite these advances, the current state-of-the-art methods model only static scenes. One reason for this is that in typical datasets there only exists one view of any dynamic scene object, e.g. a person or car. Hence, current techniques are not able to determine the 3D position of such objects. This situation is regarded to have a highly limited potential for reconstruction as stated by Park et al. [23] and Valmadre et al. [29].

We propose a technique to determine the 3D geometry of common dynamic object paths from temporally uncorrelated images, i.e. a set of images along a street with pedestrians walking (see Figure 1(a)). The challenges in this kind of datasets are that each instance of the object is typically only seen once in the images. Moreover, there is no temporal consistency between the observations of the different images and the capture times of the images are typically unknown. The only constraint available for our reconstruction is the fact that all observed instances of an object move along a common compact path in the 3D scene, which we call an object class trajectory. To obtain the object class

D. Fleet et al. (Eds.): ECCV 2014, Part VII, LNCS 8695, pp. 599–614, 2014.

(a) (b)

Fig. 1. Left: 3 images of the pedestrian dataset and the output of SFM. Right: The reconstruction of two pedestrians that are captured in the single image.

trajectory our method needs to simultaneously determine the sequence of the objects along the 3D path and the 3D positions of the objects on the path. Accordingly, this can be seen as a joint object class sequencing and trajectory triangulation, which generalizes the well known sequencing problem [2,3] and the trajectory triangulation problem [29,39,23] into a common framework. In fact, our proposed framework handles both of these problems as special cases. The resulting reconstructed object class trajectory then allows us to solve the generally ill-posed 3D reconstruction of a dynamic object from a single image by constraining the reconstruction through the 3D path of the object class. An example of a single view reconstruction of two pedestrians is shown in Figure 1(b), visualizing a generic person at the correct 3D position in the reconstruction of the 3D scene.

2 Related Work

The joint object class sequencing and trajectory triangulation is closely related to 3D reconstruction from single image, which is inherently ambiguous and difficult without further assumptions. Saxena et al. [25] propose a method for reconstruction from a single image. They compute reasonable depthmaps from a single still image by using a hierarchical multi-scale Markov Random Field (MRF) that incorporates several features. The parameters of the MRF model are trained using ground truth depth. In man-made scenes with mainly orthogonal facades (called a Manhattan world [6]), 3D reconstruction from a single image can be simplified to finding 3D lines and planes within the scene. Delage et al. [9] use a MRF model to identify the different planes and edges in the scene, as well as their orientations. Then, an iterative optimization algorithm is applied to infer the planes' positions. Ramalingam et al. [24] reconstruct the 3D lines in a Manhattan scene from an image using linear programming that identifies a sufficient minimal set of least-violated line connectivity constraints. In contrast, our joint object class sequencing and trajectory triangulation targets the reconstruction of the dynamic scene parts, in particular the object class trajectory, from a set of images.

Non-rigid structure from motion (NRSFM) methods are another class of methods related to our joint object class sequencing and trajectory triangulation. They aim to recover a deforming object's structure as well as camera motion given corresponding 2D points in a sequence of images. Tomasi and Kanade [28] propose to do rigid structure from motion through matrix factorization under an affine camera assumption. As an important extension of the well-known Tomasi-Kanade factorization, the work by Bregler et al. [5] tackles the NRSFM problem by assuming an object can be represented by a linear combination of low-order shape bases. Due to the fact that the shape bases are not unique, Xiao et al. [35] proves that using only rotation constraints results in ambiguous and invalid solutions. To solve this shape ambiguity, most existing works rely on different prior knowledge specific to the problem at hand. Not until very recently, Dai et al. [7] solved the problem by introducing a prior-free method. All these methods require a certain amount of points to be available for each frame to form a shape, as their approaches require the shape to be present. In contrast, our method only requires a single point per object class instance to infer the object class trajectory and does not have any assumption about the object shape. As a dual method, Akhter et al. [1] proposed that the smooth trajectory of each point can be restricted to a low-dimensional subspace and represented by a linear combination of Discrete Cosine Transform (DCT) bases. In contrast to [7] which requires no temporal information, [1] fails completely if frames are randomly shuffled. Whereas our proposed joint object class sequencing and trajectory triangulation does not require any temporal frame information or ordering.

Park et al. [23] reconstruct the 3D trajectory from a monocular image sequence using SFM for camera registration. They represent the trajectory by a number of low-order DCT bases, similar to Akhter et al. [1]. Their method recovers accurate 3D trajectory, but with two major flaws as pointed in [29]: (1) The user needs to manually determine the number of bases for each image sequence, and (2) the accuracy of 3D trajectory reconstruction is fundamentally limited by the correlation between the trajectory of 3D points and the motions of camera centers. This high correlation of object and camera motion is commonly occurring in real captures and degrades the reconstruction results. Valmadre et al. [29] recover the trajectory by minimizing the response of the trajectory to a set of high pass filters. Their method, in contrast to Park et al. [23], requires no basis size but still suffers under the correlation between object and camera motion. Zhu et al. [39] first estimate the 3D coordinates of a few keyframes in the video sequence, and then use those key frames to constrain the 3D trajectory. All of these methods require smooth trajectories and a given temporal order of the captured frames, while our method does not need to know the temporal order in order to successfully recover the object class trajectory.

Recently Basha et al. [2,3] propose two methods that determine the temporal order photos taken by a set of cameras. In [2], they compute the partial orders for a subset of the images by analyzing the dynamic features in the subsets 2D images. The method inherently relies on two images taken from the same static camera to eliminate the uncertainty in the sequencing. Later Basha et al. [3]

propose to enforce the constraint of a known order for the images taken by each camera. Our proposed method does not need any knowledge about the image order nor does it require multiple images of a static camera.

As another class of reconstruction, 3D articulated object reconstruction given monocular image sequences has received much attention. Several particle filter approaches have been developed for 3D human tracking [27,26]. Wei et al. [32] and Valmadre et al. [31] reconstruct the 3D human poses from a small number of 2D point correspondences obtained from uncalibrated monocular images. Based on [23], Park et al. [22] reconstruct 3D articulated motion with the constraint that a trajectory remains at a fixed distance with respect to a parent trajectory. This improves the reconstructibility over their earlier approach [23], but involves solving an NP-hard quadratic programming problem. Valmadre et al. [30] develop a dynamic programming approach which scales linearly in the number of frames to overcome solving the quadratic programming problem. While the articulated motions require multiple characteristic points observed on the same object instance, our method successfully recovers the object class trajectory from observing different object instances that each determines only a single characteristic point.

3 Joint Object Class Sequencing and Trajectory Triangulation

We now detail our method for joint object class sequencing and trajectory triangulation from uncontrolled image captures, which in particular removes the constraint of known temporal camera information and known object position. To perform joint object class sequencing and trajectory triangulation from the uncontrolled image set, we proceed as follows

1. Spatially register the cameras to a common 3D coordinate system.
2. Detect object instances and estimate motion tangents from input imagery.
3. Leverage the image positions of the object instances to simultaneously
   (a) Determine a camera ordering compliant with a continuous motion of the objects along a trajectory.
   (b) Triangulate the geometry of the corresponding motion path.

Our main contribution is an algorithm for tackling challenge 3, while we exploit known methods to solve for camera registration, object detection and motion tangents in the images. Next, we introduce the above components in more detail.

3.1 Spatial Registration

The goal of the initial spatial registration in our method is to establish camera registration in a common coordinate system. Given that in all our datasets a fair portion of the images contains static background structure, we use the publicly available structure from motion tool VisualSFM by Wu [34]. It produces the camera registration and the camera calibration. See Fig. 1(a) for an example.

The obtained camera registration determines the camera center $\tilde{\mathbf{C}}_j$ of the j-th camera. Please note, we use bold font letters $\mathbf{x}$ to indicate that an entity is a vector and regular letters x for scalar values. With the known camera calibration and registration, each pixel in the camera defines a ray direction $\mathbf{r}$ in the 3D scene space. For our object class trajectory we are only interested in the ray directions $\mathbf{r}_i$ for the different object i of the desired class (for simplicity we refer in the paper to them as objects) with $i = 1, \ldots, N$, where N is the number of detected object class instances over all frames. Hence, we only aim to compute the ray directions $\mathbf{r}_i$ for pixels belonging to the different detected objects i. The ray $\mathbf{X}_i$ in the 3D space represents a 1D subspace on which the imaged object has to lie and is described by:

$$\mathbf{X}_i(t_i) = \mathbf{C}_i + t_i \cdot \mathbf{r_i}, \tag{1}$$

where $t_i \geq 0$ is the positive distance from the camera center $\mathbf{C}_i$ along the ray $\mathbf{X}_i$. In the remaining of the paper, we keep the condition $t_i \geq 0$ implicit for the purpose of concise formulation. We denote the camera centers as $\mathbf{C}_i$ with $\mathbf{C}_i = \tilde{\mathbf{C}}_j$, where $\mathbf{C}_i$ is the center of the camera j in which the object instance i is detected. Please note that if more than one object is detected in camera j, there will be multiple $\mathbf{C}_i$ with identical positions. The unknown true distance of the object instance i along ray $\mathbf{X}_i$ is denoted as $\hat{t}_i$. Once obtained, it determines the 3D object position $\hat{\mathbf{X}}_i$.

3.2 Object Detection and Motion Tangent Estimation

Our proposed joint object class sequencing and trajectory triangulation leverages the motion tangent of the object instances, which is defined as the moving direction of the dynamic object in the 3D space, so both the objects detection and motion tangents estimation are performed based on a single image. For trajectory triangulation this has historically been solved by using videos for estimating the motion tangents [37], but for our proposed joint object class sequencing and trajectory triangulation problem, temporal coherence or temporal proximity of the images cannot be assumed. Hence only motion tangent estimation based on a single image is applied. The particular choice of object and motion tangent estimation method depends on the specific object class and the scenes. We discuss our particular choices in Section 4, and for now we assume available the positions on each image defining the rays $\mathbf{X}_i$ and a coarse estimate of the motion tangent $\mathbf{d}_i$ of object i. We determine the image positions of each detected object $i = 1, \ldots, N$ by the center of the bounding box. These object detections provide us a ray $\mathbf{X}_i(t_i)$ for each object observed in a camera, with the ray describing the one-dimensional subspace in which the detected object can be placed in the 3D scene space.

3.3 Object Class Trajectory Estimation Problem

Assuming known rays $\mathbf{X}_i(t_i)$ and the motion tangents $\mathbf{d}_i$, we will now define the object class trajectory estimation problem before delving into our data representation and our estimation framework. For the ease of description, we directly

leverage the rays $\mathbf{X}_i(t_i)$ of the detected objects i and do not use the camera registration directly as the latter is implicitly present in the ray.

Intuitively, an object class trajectory describes the motion along a path taken by the observed objects of the desired class through the 3D scene. A path can in general be any continuous curve in the 3D scene space. Since we only have a finite number of observations of objects along the path, we only sample a discrete set of 3D points on the path. The samples along the path are the 3D object positions $\hat{\mathbf{X}}_i$. Here, we represent the path as a combination of piecewise linear functions in between the sampled object positions $\hat{\mathbf{X}}_i$. The desired object class trajectory is the path of minimal length and it can be determined through a minimization of the cost function:

$$\min_{\mathbf{p}} \sum_{(i,j)\in\mathbf{p}} \|\hat{\mathbf{X}}_i - \hat{\mathbf{X}}_j\|_2^2 \tag{2}$$

with $\mathbf{p}$ representing the adjacency of the points defining the topology of the path, which is list of adjacency relationships between all the points $\hat{\mathbf{X}}_i$, $i = 1, \ldots, N$. While the trajectory above is based on the observed 3D object positions $\hat{\mathbf{X}}_i$, we can only observe the rays $\mathbf{X}_i(t_i)$. To determine the object class trajectory, we also need to determine the position of each object i along its viewing ray $\mathbf{X}_i(t_i)$, which defines the 3D position of the object $\hat{\mathbf{X}}_i$. We propose to find the adjacency relation by optimizing over variables $\mathbf{t} = [t_1, \ldots, t_N]$ and $\mathbf{p}$ jointly as follows,

$$\min_{\mathbf{p},\mathbf{t}} \sum_{(i,j)\in\mathbf{p}} \|\mathbf{X}_i(t_i) - \mathbf{X}_j(t_j)\|_2^2. \tag{3}$$

Given the motion tangents $\mathbf{d}_i$ estimated from the images, we can further constrain the trajectory, obtaining an optimization problem:

$$\min_{\mathbf{p},\mathbf{t}} \sum_{(i,j)\in\mathbf{p}} \|\mathbf{d}_{i,j} \times (\mathbf{X}_i(t_i) - \mathbf{X}_j(t_j))\|_2^2 + \lambda \|\mathbf{X}_i(t_i) - \mathbf{X}_j(t_j)\|_2^2, \tag{4}$$

where the operator $\times$ is the cross product, λ is a positive weight (discussed at length in Sec. 3.6). The direction $\mathbf{d}_{i,j}$ is selected from $\mathbf{d}_i$ and $\mathbf{d}_j$, as the motion tangent that is closest to the 3D motion direction $\mathbf{X}_i(t_i) - \mathbf{X}_j(t_j)$. In Eq. (4), the penalty of the first term increases if the direction of the vector $\mathbf{X}_i(t_i) - \mathbf{X}_j(t_j)$ deviates from $\mathbf{d}_{i,j}$. The optimization procedure simultaneously determines both the adjacency $\mathbf{p}$ of the rays and the positions of the objects through $\mathbf{t}$.

Traditionally, these problems have been treated separately as either a sequencing problem, where the 3D points are given and only the sequence of traversal needs to be estimated, or as a trajectory triangulation problem [23,29], where the sequence of observations for the trajectory is given and the 3D points along the trajectory need to be determined. Our proposed method generalizes these problems into a common framework to allow the simultaneous estimation of the adjacency of observations and the 3D position of the observations. In order to optimize Eq. (4), we propose a new discrete-continuous optimization strategy through the Generalized Minimum Spanning Tree (GMST).

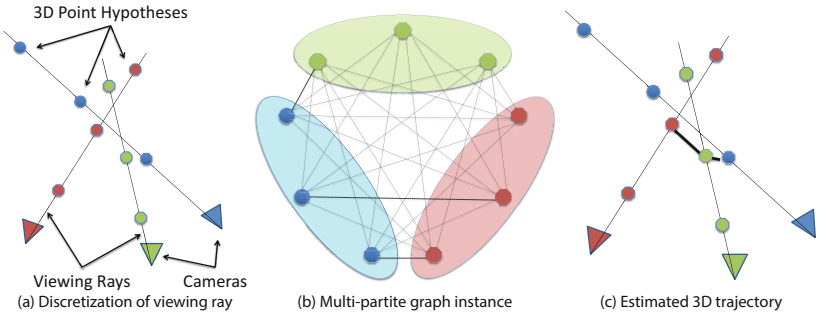


Fig. 2. Illustration of GMST. See the text for more details

3.4 Generalized Trajectory Graph

To determine the object class trajectory, we conceptually have to choose for each ray $\mathbf{X}_i(t_i)$ the 3D point, and simultaneously determine the adjacency $\mathbf{p}$ representing the adjacency relations of the rays $\mathbf{X}_i(t_i)$, which defines the topology of the object class trajectory. Our discrete-continuous optimization strategy first uses a Generalized Minimum Spanning Tree (GMST) to find adjacency $\mathbf{p}$, and followed by a convex optimization over $\mathbf{t}$ with $\mathbf{p}$ being fixed.

In the discrete optimization step, we map the continuous problem of finding the 3D point along each ray to a discrete problem of selecting a 3D point out of a set of discrete 3D points. Then we determine one 3D point along each ray and the adjacency $\mathbf{p}$ by computing the (GMST) on an undirected multipartite graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ [20]. This allows us to simultaneously determine the topology and the discrete 3D object position.

An undirected multipartite graph is a graph whose vertices are partitioned into N partite sets $\{V_1, \ldots, V_N\}$ with $|V_i| = k$, while fulfilling $\mathcal{V} = V_1 \cup V_2 \cup \cdots \cup V_N$ and $V_o \cap V_p = \emptyset, \forall o \neq p$, with $o, p \in \{1, \ldots, N\}$. The multipartite graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ has only edges between the different partite sets of vertices V_o, and all edge cost are non-negative. Now we will detail on how we define the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ for our object class trajectory estimation problem.

Each ray $\mathbf{X}_i(t)$ defines a one dimensional constraint on the 3D position of the object. We discretize the ray to obtain a discrete set of potential depth estimates. This leads to a finite set of possible 3D positions along the ray (see Figure 2(a) for an illustration), defining a finite set of 3D point hypotheses $\{\hat{\mathbf{X}}_i^o | o = 1, \ldots, k\}$, where k is the number of the discrete hypotheses along the ray. In our representation, each 3D point $\hat{\mathbf{X}}_i^o$ establishes node $V_{o,i}$ in the graph. The set of nodes $\{V_{o,i} | i = 1, \ldots, N\}$ related to the ray $\mathbf{X}_i(t)$ of object i defines a partite set of nodes V_o in the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. Given that no nodes within a group have any connecting edges, it enforces that traversing the graph to obtain an object class trajectory, the path cannot move along a ray. This is consistent with the understanding of moving between the different instances of the object class in the scene to determine the object class trajectory.

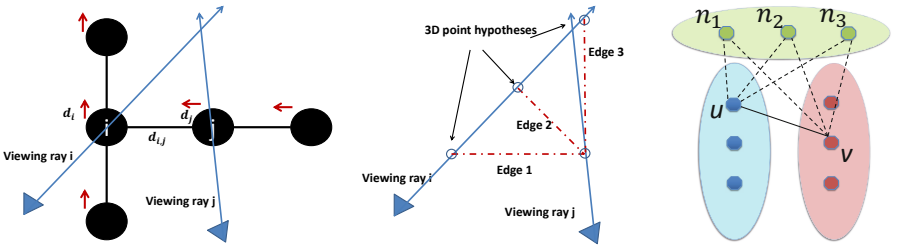


Fig. 3. In Fig. 3(a), the black nodes shows the real positions of dynamic objects. The red vector represents the direction associated with each object. In the shown example, $\mathbf{d}_{i,j}$ equals $\mathbf{d}_i$.

We now define the edge cost of the multipartite graph based on Eq. (4). The multipartite graph only has edges between the nodes from different partite sets. We define the edge direction $\mathbf{d}_{i,j}$ between any two nodes $V_{o,i}$ and $V_{p,j}$ in the partite set i and partite set j respectively, as the consistency of the 3D motion with the motion tangents $\mathbf{d}_i$ and $\mathbf{d}_j$ (see Sec. 3.2). This definition comes from the intuition that the edge direction should be compliant with the motion tangent observed in the images. Given the motion of two objects i and j and their respective motion tangents $\mathbf{d}_i$ and $\mathbf{d}_j$, it is clear that the motion between the points $\mathbf{X}_i^o$ and $\mathbf{X}_j^p$ (associated with the nodes $V_{o,i}$ and $V_{p,j}$) should be close in direction to at least one of the motion tangents $\mathbf{d}_i$ and $\mathbf{d}_j$. Therefore, we define the edge cost $e(V_{o,i}, V_{p,j})$ of the edge between the nodes $V_{o,i}$ and $V_{p,j}$ as

$$e(V_{o,i}, V_{p,j}) = \min(\|\mathbf{d}_i \times (\mathbf{X}_i^o - \mathbf{X}_j^p)\|_2^2, \|\mathbf{d}_j \times (\mathbf{X}_i^o - \mathbf{X}_j^p)\|_2^2) + \lambda \|\mathbf{X}_i^o - \mathbf{X}_j^p\|_2^2. \quad (5)$$

If only considering the first term in Eq. (5), edges with 3D motion directions that are approximately parallel to $\mathbf{d}_i$ or $\mathbf{d}_j$ have lower edge cost than 3D motion directions that are at an angle to both $\mathbf{d}_i$ and $\mathbf{d}_j$. For instance, Edge 1 and Edge 3 in Fig. 3(b) have a relatively lower cost than Edge 2 because Edge 1 is parallel to $\mathbf{d}_j$ and Edge 3 is parallel to $\mathbf{d}_i$.

3.5 GMST

A Generalized Minimum Spanning Tree (GMST) on the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is a tree of minimal cost that spans exactly one node from each partite set V_i. For our proposed graph, it means a GMST includes exactly one 3D point from each ray. Furthermore, GMST prefers the edge $((o,i),(p,j))$ that has small $\|\mathbf{X}_i^o - \mathbf{X}_j^p\|^2$ and is compliant with the motion tangents in the images, as those edges have lower edge cost. Accordingly, a GMST is our desired solution for estimating the object class trajectory. Notice that if we sample infinite number of 3d points along each viewing ray, the GMST problem is equivalent to the original cost function Eq. (4).

The multipartite graph defined above contains a large amount of edges, which increases the complexity of computing the GMST. We use a deterministic way introduced by Ferreira et al. [13] to remove those edges that are guaranteed not to be included in the GMST. Here we use a specific toy example in Fig. 3(c) to illustrate the method. If the cost of edge (u, v) is larger than any cost of the 6 edges (u, n_l) and (v, n_l), $l = 1, 2, 3$, the edge (u, v) is safe to be removed. A simple proof is that if edge (u, v) exists in the computed GMST, we could remove edge (u, v) and replace it by one of the 6 edges to obtain a new GMST with lower cost. Therefore, edge (u, v) can not be present in the GMST. Moreover, it is plausible to explore other ways to remove edges based on given prior information. For instance, if it is known the pairwise neighboring 3D objects are close in 3D space, we can safely remove the edges that connects two farther point hypotheses by applying a threshold.

The GMST problem was first introduced by Myung et al. [20] and was extensively studied in the past two decades [12,20,21,13,10] due to its wide applications in telecommunications, agriculture watering, and facility distribution design [20,10]. Unlike the minimum spanning tree (MST) problem, which can be solved in polynomial time, finding the GMST is proved to be NP-hard [20]. Myung et al. [20] and Feremans et al. [12] propose several integer programming formulations for the GMST problem. However, those provide no guarantee of efficiency, especially when the problem scale is large. The computational challenge of the GMST problem has led to the development of metaheuristics [21,13] that search the hypothesis space, and are empirically shown to be effective.

We exploit the state-of-the-art GRASP-based approach proposed by Ferreira et al. [13]. GRASP (Greedy Randomized Adaptive Search Procedure) is a metaheuristic that consists of iterations made up two phases: 1) solution construction and 2) solution improvement through local search. Ferreira et al. [13] proposed the method considering several solution construction algorithms, a local search procedure, and two additional mechanisms: path-relinking and iterative local search. We refer readers to their paper [13] for more details.

The output of GMST computation is the estimation of the 3D points $\widehat{\mathbf{X}}$ and the adjacency topology of the object class trajectory. Then $\mathbf{d}_{i,j}$ equals one of $\mathbf{d}_i$ and $\mathbf{d}_j$ that has smaller angle to the vector $\widehat{\mathbf{X}}_i - \widehat{\mathbf{X}}_j$,

$$\mathbf{d}_{i,j} = \underset{\mathbf{d} \in \{\mathbf{d}_i, \mathbf{d}_j\}}{\operatorname{argmax}} (|\mathbf{d} \cdot (\widehat{\mathbf{X}}_i - \widehat{\mathbf{X}}_j)|). \tag{6}$$

We fix the adjacency $\mathbf{p}$ given by the GMST, and add a final continuous refinement step for the 3D object position $\widehat{\mathbf{X}}_i$, through a convex program optimization over variable $\mathbf{t}$

$$\min_{\mathbf{t}} \sum_{(i,j) \in \mathbf{p}} \|\mathbf{d}_{i,j} \times (\mathbf{X}_i - \mathbf{X}_j)\|_2^2 + \lambda \|\mathbf{X}_i - \mathbf{X}_j\|_2^2 \tag{7}$$

3.6 Reconstructability Analysis

Now we analyze the reconstructability of the proposed method, i. e. determining under which conditions the solution of Eq. (4) generates accurate 3D points.

The direct analysis of Eq (4) is difficult, as it needs to determine in which situation the adjacency $\mathbf{p}$ with minimum cost, out of N^{N-2} possible adjacencies ([33]), corresponds to the object class trajectory. We find that having the motion tangent constraint reduces the possibility of finding the wrong adjacency $\mathbf{p}$. Hence, we focus on the reconstructability of the continuous method Eq. (7) given the adjacency $\mathbf{p}$.

Assume we already know the ground truth 3D point $\mathbf{X}_i^*$ of object i, $i = 1, \ldots, N$. Given that $\mathbf{X}_i^*$ is present on the viewing ray $\mathbf{X}_i$, we move the camera center $\mathbf{C}_i$ to $\mathbf{X}_i^*$ along the ray $\mathbf{X}_i(t)$ in direction $\mathbf{r}_i$. Then any point on the line that passes through $\mathbf{X}_i^*$ and has ray direction $\mathbf{r}_i$ can be represented as $\mathbf{X}_i(s_i) = \mathbf{X}_i^* + s_i \cdot \mathbf{r}_i$, where s_i is the signed distance (not the positive distance as defined by the t_i). Then Eq. (7) can be reformulated as:

$$\min_{\mathbf{s}} \sum_{(i,j)\in\mathbf{p}} \|\mathbf{d}_{i,j} \times (\mathbf{X}_i(s_i) - \mathbf{X}_j(s_j))\|_2^2 + \lambda \|\mathbf{X}_i(s_i) - \mathbf{X}_j(s_j)\|_2^2, \tag{8}$$

where $\mathbf{s} = [s_1, \ldots, s_N]$. Though s_i is signed distance and t_i is positive distance, minimizing Eq. (7) and Eq. (8) still output the same 3D point positions, as long as the computed 3D points in Eq. (8) are in front of the camera centers. We will see that this is normally true, because the computed 3D points are typically close to their ground truth position if the system is well-conditioned.

We denote the solution of Eq. (8) as $\mathbf{s}^{\text{opt}}$. The true 3D points are ideally reconstructed if $\mathbf{s}^{\text{opt}} = 0$, since $\mathbf{X}_i(0)$ equals to $\mathbf{X}_i^*$ given $\mathbf{s}^{\text{opt}} = 0$. More specifically, $\mathbf{s}^{\text{opt}}$ equals the signed Euclidean distance between the 3D points produced by Eq. (7) and the ground truth X_i^*. Therefore, $\|\mathbf{s}^{\text{opt}}\|$ is the error of the estimated 3D points by Eq. 7. In the remaining of this section, we further analyze when $\|\mathbf{s}^{\text{opt}}\|$ is small to understand the quality of the estimated 3D points.

The minimum value of Eq. (8) is achieved at the point where the first derivative relative to $\mathbf{s}$ equals 0. This produces a linear equation system $\mathbf{A}\mathbf{s}^{\text{opt}} = \mathbf{b}$, where the ith row and jth column of matrix $\mathbf{A}$ is

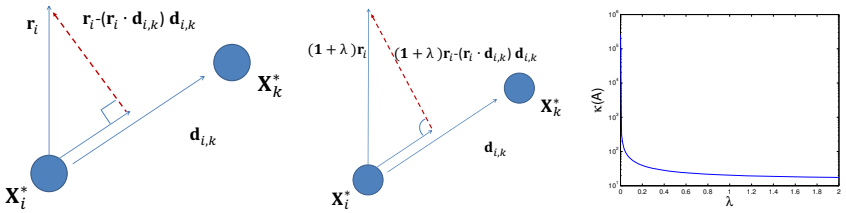
$$A_{ij} = \begin{cases} [(\mathbf{r}_i \cdot \mathbf{d}_{i,j})\mathbf{d}_{i,j} - (1+\lambda)\mathbf{r}_i] \cdot \mathbf{r}_j & \text{if } i \neq j \text{ and } (i,j) \in \mathbf{p} \\ 0, & \text{if } i \neq j \text{ and } (i,j) \notin \mathbf{p} \\ \sum_{(i,k)\in\mathbf{p}} [1 + \lambda - (\mathbf{r}_i \cdot \mathbf{d}_{i,k})^2] & \text{if } i = j \end{cases} \tag{9}$$

The ith element of vector $\mathbf{b}$ is

$$b_i = \sum\nolimits_{(i,k)\in\mathbf{p}} (\mathbf{X}_k^* - \mathbf{X}_i^*) \cdot [(1+\lambda)\mathbf{r}_i - (\mathbf{r}_i \cdot \mathbf{d}_{i,k})\mathbf{d}_{i,k}] \tag{10}$$

Eq. (9) and Eq. (10) have the following interesting properties:

1. If $\mathbf{b}$ is 0, $\mathbf{s}^{\text{opt}}$ equals 0, which means the solution of Eq. (7) recovers the true 3D points. There are a few situations $\mathbf{b}$ equal 0. (1) In the case of a static object $\mathbf{X}_i^* = \mathbf{X}_k^*$, $\mathbf{b}$ equals 0 based on Eq. (10). (2) Careful observation reveals that if λ is set to 0, in Eq. (10) the vector $(1+\lambda)\mathbf{r}_i - (\mathbf{d}_{i,k} \cdot \mathbf{r}_i)\mathbf{d}_{i,k}$ is perpendicular to vector $\mathbf{X}_i^* - \mathbf{X}_k^*$ (Fig. 4(a)), hence $b_i = 0$. However, we



(a) $\lambda = 0$ (b) $\lambda > 0$ (c) $\kappa(\mathbf{A})$ versus λ

Fig. 4. Left two: Plot of Eq. (10) with $\lambda = 0$ and $\lambda > 0$. Right: $\kappa(\mathbf{A})$ given different λ.

will show that with $\lambda = 0$, the linear system $\mathbf{As} = \mathbf{b}$ is unstable due to the high condition number of $\mathbf{A}$. (3) As shown in Fig. 4(b), as λ increases from 0, the two vectors slowly deviate from being perpendicular. Therefore, b_i is likely to be small if λ is close to 0.

2. Since we can not control 3D positions and there are typically small measurement errors in $\mathbf{d}_{ij}$, $\mathbf{b}$ does not exactly equal to 0. This can be regarded as a small disturbance of $\mathbf{b}$ around $\mathbf{0}$. For the linear system $\mathbf{As}^{\text{opt}} = \mathbf{b}$, one can think of the condition number $\kappa(\mathbf{A})$ as being (roughly) the rate at which the solution, $\mathbf{s}^{\text{opt}}$, will change with respect to a change in $\mathbf{b}$. $\kappa(\mathbf{A})$ is available as it solely depends on $\mathbf{r}_i$, $\mathbf{d}_{i,j}$ and λ, but not on the ground truth 3D points $\mathbf{X}^*$. Therefore, we can roughly estimate the reliability of the reconstructed 3D points by computing $\kappa(\mathbf{A})$. Moreover, we empirically found that the condition number of matrix $\mathbf{A}$ is inversely related to λ. The condition number shown in Fig. 4(c) is computed using 100 random cameras, and averaged over 200 trials. We can see $\kappa(\mathbf{A})$ is large if λ is close to 0 and drop dramatically with small λ. Then $\kappa(\mathbf{A})$ decreases monotonically and slowly as λ increases. In our experiments, we choose $\lambda = \frac{1}{15}$ as a balance of having good chance of small $\mathbf{b}$ and the stability of the linear system.

In conclusion, if the adjacency $\mathbf{p}$ is correctly found, the reconstructabililty of the object class trajectory mainly depends on the condition number of the linear system. Given the well-conditioned system and correct motion tangent $\mathbf{d}_{i,j}$, we are able to reconstruct the 3D positions close to the ground truth.

4 Object Detector and Motion Tangent Estimation

Before presenting our experimental evaluation, we first briefly describe the particular object detectors we use in our experiments. Single image based object detection is a well studied problem in computer vision with a wide variety of method readily available [36,8,11]. Similarly, there is a large number of motion tangent estimation methods in the literature [4,16,17,19]. We opt for leveraging the method that jointly determines the face position and its motion tangent direction [38]. In our experiments, the detection threshold is set to -0.35 to avoid false detections, as the false alarm may disturb our algorithm. Our chosen

Fig. 5. Detected objects and estimated motion tangents using different detectors

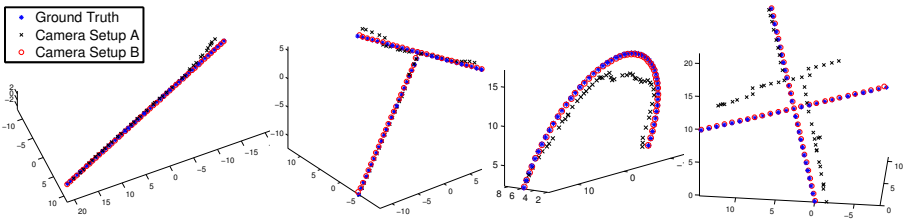


Fig. 6. Example results for line path, T-junction path, half circle and crossed paths

detectors provide a motion tangent of object i that is quantized every $\theta = 15°$ in the range of $-90°$ and $90°$.

For cars and pedestrians with small faces in the image, we default to the deformable parts detector [11,15]. We used the pre-trained model with detection threshold 0.35. The moving directions of the pedestrians and cars are estimated using the 3D point cloud (output of VisualSFM) of the background wall by assuming the dynamic objects move parallel to the wall. This is normally true for the Manhattan Scene. Some of the detection results are shown in Fig. 5.

5 Experiments

We evaluate our algorithm on both synthetic and real datasets. The GMST algorithm used in our method [13] searches the hypothesis space, which stops either the GMST cost is under a preset value or the run time reaches a preset limit. For all experiments, we use the time limit to stop searching, given the lack of an adequate *a priori* approximation of the true GMST cost for each dataset.

Table 1. The table shows the average errors. The subscript represents camera setup. The absence of asterisk represents the GMST algorithm output, and the asterisk is the refined output of Eq. (7). Notice that in ground truth 3D points, the average distance between every pair of nearest points equals 1.

	single line	T junction	double lines	half circle	sine wave	cross
error_A	0.5963	1.9688	1.5169	2.3751	2.3705	3.4111
error_A^*	0.4263	1.9148	1.4982	2.3340	2.3516	3.4030
error_B	0.2151	0.2126	0.7824	0.2281	0.2578	0.2251
error_B^*	0.0287	0.0944	0.7692	0.1074	0.2305	0.1308

Synthetic Datasets. Our first experiment uses synthetic data, with six different object class trajectory shapes on a plane, including a single line path, a T-junction path, a path with two parallel lines, a half circle path, sine wave shaped path, and two crossed path. For each of these shapes, we tested 32 instances, with each containing 50 randomly chosen object observation at different 3D positions. For more convenient error estimation during the evaluation, we normalized the 3D points so that the average distance between every pair of nearest points equals 1. The cameras are randomly generated around the 3D object points with two different configurations. In camera configuration A, all the camera centers stay in the same plane as the 3D points, which is more difficult since each viewing ray may intersect the ground truth path several times. In camera configuration B, the camera centers are set randomly off the plane, with the angle between the viewing ray and the plane being at most 10° and camera distance of 2-3 times the length of the path. We choose $k = 100$ uniformly distributed discrete 3D hypotheses $\mathbf{X}_i^o$ along each viewing ray $\mathbf{X}_i$ in a range that contains the ground truth 3D point. The size of the range is set as 1.5 times the length of the path. Notice that while the ground truth 3D points lie in the range, it is not guaranteed to be one of the discrete samples $\mathbf{X}_i^o$.

Errors are measured using the Euclidean distance between the estimated 3D points and the ground truth. The average errors over the 32 instances for each shape category are listed in Table 1. We report errors of the GMST output, and the errors after the continuous refinement using Eq. (7). Table 1 shows our continuous refinement always improves the reconstruction accuracy over the GMST approximation. The results demonstrate *off-plane* cameras yield improved results than *in-plane* cameras for complex paths (e.g. crossed paths), due to the multiplicity of ray-to-path intersections. In these cases the GMST solution has a more complex search space and yields a sub-optimal solution. However, the condition number of the linear system does not vary significantly across configurations. Fig. 6 shows the estimated 3D points overlaid onto the ground truth.

Real Datasets. We evaluated our method on two image datasets registered by VisualSFM [34]. The detection confidence threshold is set high in order to lower down the false alarm rate. However, a very small amount of false alarms are purged manually as it may affect the reconstruction. We sample 100 samples along the viewing ray in the range $[0, far]$, where far is estimated using the model scale. The running time for each object class trajectory is set to 3 hours.

The first dataset captures random pedestrians walking on the sidewalk, and random cars running on a lane. It contains 135 images with 82 valid car detections and 137 valid pedestrian detections. The scene and the reconstructed object class trajectory are shown in Fig. 7. The second dataset captures several people who are walking on a T-junction shaped path at the corner of a building. It contains 47 images with 66 valid detections. Using the camera positions, we convert the face directions into the global coordinate system to obtain the motion tangents $\mathbf{d}_i$ of the moving people. For illustration, we construct the background static scene using CMPMVS [18]. The general 3D human and car mesh models

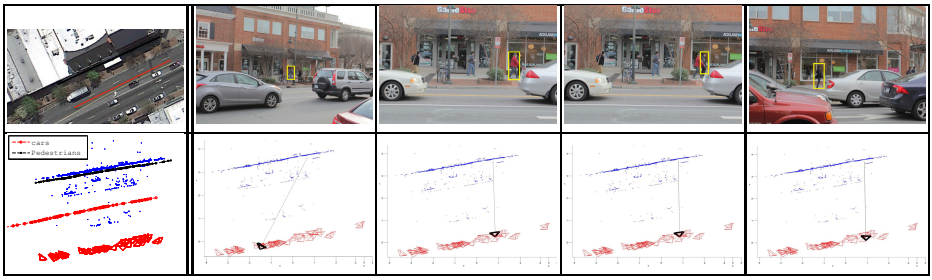


Fig. 7. The left column: an aerial image showing the scene and a figure showing the cameras and reconstructed cars and pedestrians. The right four columns: four pedestrian detections (shown in yellow rectangles) and the poses of the corresponding cameras. These four pedestrians are adjacent in the reconstructed object class trajectory. Notice that the second and the third images are the same image but with different detections.

Fig. 8. Two views for each of the reconstructed results

are inserted into each of the estimated 3D positions. We show different views of the reconstructed results in Fig. 8.

6 Conclusions

We proposed a solution to the novel joint object class sequencing and trajectory triangulation problem, which allows the reconstruction of an object class trajectory from unordered images for which the capture times are unknown and there is no requirement of more than one view observing any object instance. This problem has previously been seen as highly limited in reconstructabililty. We evaluated our proposed method on synthetic and real world datasets and show promising results demonstrating the feasibility of the proposed approach to solve the joint object class sequencing and trajectory triangulation problem and in fact the first time demonstrating its solvability.

Acknowledgement. This material is based upon work supported by the National Science Foundation under Grant No. IIS-1252921 and No. IIS-1349074.

References

1. Akhter, I., Sheikh, Y.A., Khan, S., Kanade, T.: Nonrigid structure from motion in trajectory space. In: NIPS (2008)
2. Basha, T., Moses, Y., Avidan, S.: Photo sequencing. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VI. LNCS, vol. 7577, pp. 654–667. Springer, Heidelberg (2012)
3. Basha, T., Moses, Y., Avidan, S.: Space-time tradeoffs in photo sequencing. In: ICCV (2013)
4. Blanz, V., Vetter, T.: Face recognition based on fitting a 3D morphable model. PAMI (2003)
5. Bregler, C., Hertzmann, A., Biermann, H.: Recovering non-rigid 3D shape from image streams. In: CVPR (2000)
6. Coughlan, J.M., Yuille, A.L.: Manhattan world: Compass direction from a single image by bayesian inference. In: ICCV (1999)
7. Dai, Y., Li, H., He, M.: a simple prior-free method for non-rigid structure-from-motion factorization. In: CVPR (2012)
8. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR (2005)
9. Delage, E., Lee, H., Ng, A.Y.: Automatic single-image 3D reconstructions of indoor manhattan world scenes. In: International Symposium on Robotics Research (2005)
10. Dror, M., Haouari, M., Chaouachi, J.: Generalized spanning trees. European Journal of Operational Research 120(3), 583–592 (2000)
11. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part based models. PAMI (2010)
12. Feremans, C., Labbe, M., Laporte, G.: A comparative analysis of several formulations for the generalized minimum spanning tree problem. Networks (2002)
13. Ferreira, C.S., Ochi, L.S., Parada, V., Uchoa, E.: A grasp-based approach to the generalized minimum spanning tree problem (2012)
14. Frahm, J.-M., et al.: Building rome on a cloudless day. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 368–381. Springer, Heidelberg (2010)
15. Girshick, R.B., Felzenszwalb, P.F., McAllester, D.: Discriminatively trained deformable part models, release 5, `http://people.cs.uchicago.edu/~rbg/latent-release5/`
16. Gu, L., Kanade, T.: 3D alignment of face in a single image. In: CVPR (2006)
17. Jain, V., Learned-Miller, E.G.: Fddb: a benchmark for face detection in unconstrained settings. UMass Amherst Technical Report (2010)
18. Jancosek, M., Pajdla, T.: Multi-view reconstruction preserving weakly-supported surfaces. In: CVPR (2011)
19. Jones, M., Viola, P.: Fast multi-view face detection. Mitsubishi Electric Research Lab TR-20003-96 (2003)
20. Myung, Y., Lee, C., Tcha, D.: On the generalized minimum spanning tree problem. Networks (1995)
21. Oncan, T., Cordeau, J., Gilbert, L.: A tabu search heuristic for the generalized minimum spanning tree problem. European Journal of Operational Research (2008)
22. Park, H., Sheikh, Y.: 3D reconstruction of a smooth articulated trajectory from a monocular image sequence. In: ICCV (2011)

23. Park, H.S., Shiratori, T., Matthews, I., Sheikh, Y.: 3D reconstruction of a moving point from a series of 2D projections. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part III. LNCS, vol. 6313, pp. 158–171. Springer, Heidelberg (2010)
24. Ramalingam, S., Brand, M.: Lifting 3D manhattan lines from a single image. In: ICCV (2013)
25. Saxena, A., Chung, S.H., Ng, A.Y.: 3D depth reconstruction from a single still image. IJCV (2008)
26. Sidenbladh, H., Black, M.J., Fleet, D.J.: Stochastic tracking of 3D human figures using 2D image motion. In: Vernon, D. (ed.) ECCV 2000. LNCS, vol. 1843, pp. 702–718. Springer, Heidelberg (2000)
27. Sminchisescu, C., Triggs, B.: Kinematic jump processes for monocular 3D human tracking. In: CVPR (2003)
28. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography: a factorization method. IJCV (1992)
29. Valmadre, J., Lucey, S.: General trajectory prior for non-rigid reconstruction. In: CVPR (2012)
30. Valmadre, J., Zhu, Y., Sridharan, S., Lucey, S.: Efficient articulated trajectory reconstruction using dynamic programming and filters. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part I. LNCS, vol. 7572, pp. 72–85. Springer, Heidelberg (2012)
31. Valmadre, J., Lucey, S.: Deterministic 3D human pose estimation using rigid structure. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part III. LNCS, vol. 6313, pp. 467–480. Springer, Heidelberg (2010)
32. Wei, X.K., Chai, J.: Modeling 3D human poses from uncalibrated monocular images. In: ICCV (2009)
33. Wikipedia: Cayley's formula, `http://en.wikipedia.org/wiki/Cayley's_formula`
34. Wu, C.: Visualsfm: A visual structure from motion system (2011), `http://homes.cs.washington.edu/~ccwu/vsfm/`
35. Xiao, J., Chai, J.-X., Kanade, T.: A closed-form solution to non-rigid shape and motion recovery. In: Pajdla, T., Matas, J. (eds.) ECCV 2004. LNCS, vol. 3024, pp. 573–587. Springer, Heidelberg (2004)
36. Zhang, J., Marszalek, M., Lazebnik, S., Schmid, C.: Local features and kernels for classification of texture and object categories: A comprehensive study. In: CVPR (2006)
37. Zhao, W., Chellappa, R., Phillips, P.J., Rosenfeld, A.: Face recognition: a literature survey. ACM Computing Surveys, CSUR (2003)
38. Zhu, X., Ramanan, D.: Face detection, pose estimation, and landmark localization in the wild. In: CVPR (2012)
39. Zhu, Y., Cox, M., Lucey, S.: 3D motion reconstruction for real-world camera motion. In: CVPR (2011)