I. INTRODUCTION

Primary purpose of this document is for better understanding of algorithm-1 which is Multi-view Stochastic Hill-Climbing, and that of all the terms in the main objective function of the main paper. We shall start with all the terms in the main objective function and then explain Multi-view Stochastic Hill-Climbing.

II. MAIN OBJECTIVE FUNCTION

The main objective function in the main paper, equation 7, contains four terms namely \textit{boxfit}, \textit{detectionfit}, \textit{deepfit}, \textit{photofit}. These terms exploit different information and come into action at different times to tackle various problems that we faced.

A. \textit{boxfit}

Purpose of \textit{boxfit} is to estimate pose of camera with respect to object(car) for first 3-5 frames. Using this pose, we optimize for shape of the car using Multi-view Stochastic Hill Climbing. Ideally we should be able to do this for all the images. But empirically, we found out, on multiple datasets, that tracks even with state-of-the-art Deep-Match/Deep-flow [2] become noisy and don’t yield great results for pose estimation after 6-7 frames in general. Thus, after optimizing for shape of the car, we just need to optimize for pose in subsequent frames. Some more explanation about how \textit{boxfit} works is given below.

The features within 2D bounding box are divided into two sets as shown here in Figure 1 such that inliers are maximum when we compute Homography matrices to fit each of these two sets of features. The fact, that these two planes on a car are practically perpendicular, is used to eliminate incorrect combinations of Rotation and Translation matrices that we get from decomposition of the previously calculated Homography matrices. Top view of reconstruction is shown in Figure 2.

B. \textit{deepfit}

\textit{deepfit} uses 2D projection of deformable wireframe and and deep matches contained within the closed 2D space contained within projection of deformable wireframe. The principle idea here is that shear between above mentioned two entities in consecutive frames should be as minimal as possible. An example with one feature point and one vertex of deformable wireframe is presented below for understanding purpose.

Figure 3(a) and (b) are the same images at the instant \(t\). Two points are marked on it, point on the head of the arrow is a “part” of deformable wireframe, corresponding to the term \(P(S_j(\alpha), P(t))\) in equation (5) in the main paper and point on the tail of the arrow (near the number plate) is a feature detected within the area covered by deformable wireframe(This feature is matched at next instant in (c) and (d) as well), which corresponds to the term \(x^t_i\). The Yellow arrow depicts the distance between these two points. This
arrow is shown in all four images for comparison.

Figure 3(c) and (d) show different particles at instant \( t + 1 \) but they show particles with different poses. Different particles mean they have different pose parameters. The term \( x_{t+1} \) in equation (5) in the main paper is the tracked/matched point of \( x_t \) whereas the term \( P(S_j(\alpha), P(t+1)) \) corresponds to the head of the white arrow shown in (c). The white and yellow arrows almost coincide in 9(d). The deepfit term \( D(S_j(\alpha), P(t), P(t+1), x^t) \) is the difference between the magnitudes of these two arrows. Considering just one feature match and one "part", it can be inferred that (d) shows the correct particle and (c), one of the incorrect ones. It can be observed in (c) the difference between two arrows which highlight shear between the matched feature point at \( t + 1 \) and vertex of deformable wireframe. Only one feature match and one "part" is considered here just to make understanding simple. Practically, we use all the matches and all the "parts".

Fig. 3: (a) and (b) are the same images, which is the initial instant \( t \). (c) and (d) are two of the different particles (estimated poses) at the next instant \( t + 1 \), where pose in (d) is closer to the correct pose and (c) has incorrect pose.

C. detectionfit
detectionfit utilises output of local part detectors as defined and used in [3]. For each part, we get part (log-)likelihood score at every pixel in the image from a multi-class Random Forest as per equation (4) in the main paper. We sum over (log-)likelihood of all visible parts. Only the parts that are visible are considered because some parts would not be visible for given viewpoint due to self-occlusion. In order to handle this, the term \( o_j(S(\alpha)) \) in equation (4) in the main paper is a binary indicator function for visibility. Here part likelihood \( L_k \) is normalized by background likelihood \( L_b \) so that it discriminates the positive class from its background the most, following [1]. Finally the likelihood is re-normalized to the number of visible terms.

This term is used in optimizing shape as well as pose. As shown in equation (7) in the main paper, normalized score considering all visible parts is used for scoring while optimizing for shape. While in case of optimizing pose, likelihood of wheels is given quite a heavy weightage as the peaks there is much more prominent than all other parts. Such an example is shown here in Figure 5.

Fig. 5: Yellow part in this figure shows thresholded output of part detector for wheels.

D. photofit
photofit is used for estimating pose only when car is far away from the camera and hence is used when height of the car in image becomes less than 50 pixels. We use normalized 2D cross correlation of each particle with the correct one in previous frame as the measure. As mentioned in the paper, we leverage on the fact that distant objects more or less undergo affine transformations of their textured surfaces, and hence the immediate texture surrounding the corners of the wireframe models projection tend to remain intact.

III. Multi-view Stochastic Hill Climbing
This algorithm can be seen as multi-view extension of the objective function presented in [3]. Apart from a 2D bounding box and a coarse pose of car, this algorithm takes pose estimated from the term boxfit as inputs and it optimizes for shape of the car as well as gives accurate pose in the first image.

As shown in Algorithm-1 in the main paper, we generate 250 particles with shape and pose parameters similar to
[3], except the fact that the particles in [3] don’t contain information about localization of car in 3D. For every such particle we generate 400 candidates following a normal distribution. For each such candidate we perform scoring and candidate with the best score is saved. Likewise, the particle with the best score is saved as well. We perform 20 such iterations and finally the particle with the best score is chosen.

Output of local part detectors as mentioned in section II-C is used for scoring. Each visible part of deformable wireframe is projected on the image and part likelihood score of corresponding part at that corresponding pixel is considered and sum of all the visible parts is considered as the score for that candidate.

**REFERENCES**

