FEATS: Synthetic Feature Tracks for Structure from Motion Evaluation

Joseph DeGol[1], Jae Yong Lee[1], Rajbir Kataria[1], Daniel Yuan[1], Timothy Bretl[1], Derek Hoiem[1,2]
[1] University of Illinois Urbana-Champaign [2] Reconstruct Inc.
{degol2, lee896, rk2, tbretl, dhoiem}@illinois.edu derek.hoiem@reconstructinc.com

Abstract

We present FEATS (Feature Extraction and Tracking Simulator), that synthesizes feature tracks using a camera trajectory and scene geometry (e.g. CAD, multi-view stereo). We introduce 2D feature and matching noise models that can be controlled using a few parameters. We also provide a new dataset of images and ground truth camera pose. We process this data (and a synthetic version) with several current SfM algorithms and show that the synthetic tracks are representative of the real tracks. We then show two practical uses of FEATS: (1) we generate hundreds of trajectories with varying noise and show that COLMAP is more robust to noise than OpenSfM and VisualSfM; and (2) we calculate 3D point error and show that accurate camera pose estimates do not guarantee accurate 3D maps.

1. Introduction

We present FEATS (Feature Extraction And Tracking Simulator), a simulator that synthesizes feature matches (rather than images) using camera poses and scene geometry (e.g. CAD, laser, multi-view stereo) to (1) predict if 3D reconstruction on real images will succeed and (2) evaluate structure from motion (SfM) using controlled noise and perfect ground truth. While it is difficult to synthesize images that are realistic enough for evaluation [51], synthesizing geometry is easier and has been shown to be effective for training and evaluation [44]. Moreover, SfM algorithms are independent of the images given extracted and matched features (i.e., "tracks" or "match graphs"). Synthesizing feature tracks also enables direct comparison among SfM algorithms (as feature detection/matching is held constant), and we can systematically vary trajectories, feature noise, and matching outliers to better understand the strengths and weaknesses of the algorithms. Lastly, our simulator provides exact ground truth 6DoF camera pose and 3D point locations, which are crucial to measure the accuracy of SfM.

FEATS is motivated by two challenges. **The first challenge** is wanting to know if a data collection path will result in a successful reconstruction. For 3D mapping of construction sites [26, 25, 18, 19, 1, 5, 7], repeated (e.g. weekly) drone flights collect image data for 3D mapping the scene. For each flight, a path is planned using software before the drone collects the images. These images are input to an SfM algorithm which takes hours (sometimes days) to create a 3D map (which may be a failure). This process is time consuming, costly, and frustrating when the reconstruction fails. FEATS offers a solution because FEATS can use previous maps to simulate feature extraction and matching (which are input to SfM) to evaluate (and fix) planned paths prior to data collection. **The second challenge** is obtaining aligned camera pose and 3D point ground truth data (both, together) of large environments for evaluating SfM algorithms. Currently, capturing ground truth is challenging and costly (see Section 2), but necessary because (1) large environments are a common SfM application [16, 10, 21, 53, 8, 45, 48]; and (2) accurate estimates of camera pose do not guarantee accurate estimates of 3D points (see Section 5). FEATS provides ground truth camera pose and 3D point locations for arbitrarily large scenes.

We introduce a new dataset of image collections with ground truth camera pose and use FEATS to generate synthetic equivalents. We show that SfM results on these synthetic equivalents are predictive of SfM results from processing real images. We then demonstrate two new benchmarks for SfM enabled by FEATS. In particular, we vary noise parameters for synthetic tracks and show how SfM algorithms perform, and we calculate error between ground truth 3D points and reconstructed point clouds.

In summary, our **contributions** are: (1) FEATS, a simulator to synthesize feature tracks (with ground truth pose and 3D point locations) for evaluation of SfM algorithms; (2) a new dataset (images and ground truth) that focuses on current pitfall scenarios for SfM (e.g. pure rotation); (3) experiments verifying that FEATS produces synthetic tracks that represent real world data; and (4) two new evaluations of current SfM algorithms using synthetic features.

2. Related Work

Simulation and Synthetic Data: Synthetic data has become popular for solving computer vision problems be-

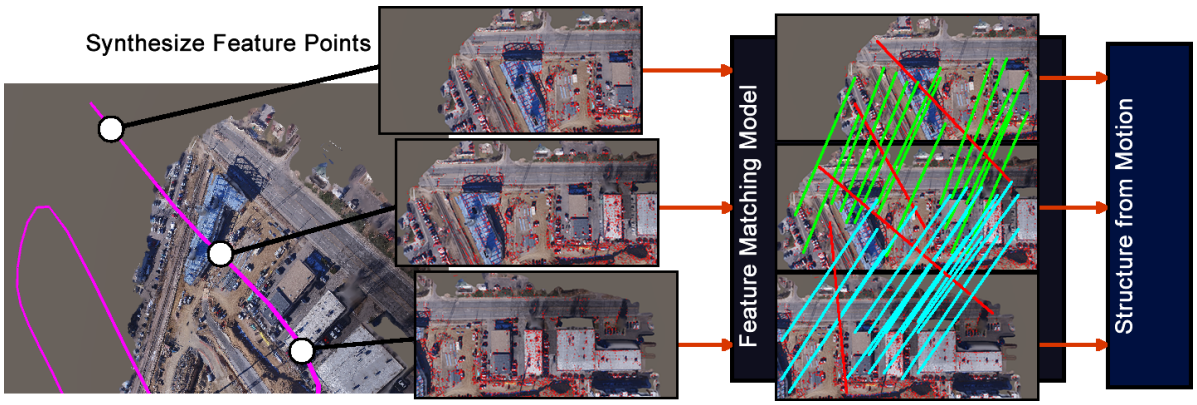


Figure 1: FEATS (Feature Extraction And Tracking Simulator) synthesizes feature tracks from camera motion through 3D scenes. The feature tracks are controlled through noise model parameters and input into SfM algorithms. We use new real world data (with ground truth) to validate that simulated tracks are predictive of real world tracks. We then show how several SfM algorithms perform for varying noise and calculate 3D point error on the resulting reconstructions.

cause of increasingly powerful computer graphics tools and the need for large amounts of ground truth data. Kaneva et al. [32] and Butler et al. [14] generate data for image feature and optical flow evaluation respectively. Battaglia et al. [11] use simulation data to study human interaction with objects. Taylor et. al. [50], Marin et al.[35], Stark et al. [47], and Hattori et al.[30] use synthetic data for training object classifiers. The main challenge is using synthetic data to achieve results that transfer to real data. Works including Gaidon et al. [23], Fisher et al. [20], Handa et al. [28, 27], Ros et al. [39], Vazquez et al. [52, 54], and Shotton et al. [44] address this challenge by training models using a mix of synthetic and real data to achieve state-of-the-art results on labeling and classification.

There are also a few simulators for 3D reconstruction. Handa et al. [29] uses two virtual 3D scenes to create RGB-D images from RGB and depth noise models. CARLA (Dosovitskiy et al. [17]) and AirSim (Shah et al. [43]) each provide a small number of highly detailed virtual 3D worlds with moving vehicles and synthetic images and depth maps. FEATS is different because it enables unlimited scenes and camera paths. Moreover, all previously mentioned work synthesizes images. Vaudrey et al. [51] shows that results on synthetic images do not easily transfer to real world results because synthetic images have crisp image boundaries and consistent pixel intensity values. Alternatively, Shotten et al. [44] shows that synthetic geometry can effectively transfer to real world results. We follow in the footsteps of this work and provide the first simulation environment that synthesizes image feature tracks.

Real 3D Reconstruction Data: Collecting ground truth camera pose and scene geometry is difficult, costly, and few datasets exist that provide both. GPS tagged images can have meter level accuracy and are not ideal for ground truth camera pose. Real Time Kinematics (RTK) GPS systems are more accurate, providing centimeter level accuracy. Datasets such as Malaga [12], Rawseeds [6], KITTI Driving [24] and Cornell Quad [16] all use RTK GPS for ground truth camera pose. However, none of these datasets provide ground truth geometry because they cover large outdoor scenes. For small workspaces, motion capture systems (MoCap) can produce millimeter accuracy ground truth camera pose. The TUM RGB-D dataset [49] and EuRoC dataset [13] provide tens of trajectories in small indoor and outdoor scenes with camera pose ground truth from MoCap. The EuRoC dataset also provides geometry ground truth using a laser scanner. Tanks and Temples [33] and ETH3D [42] use laser scanners to provide 3D geometry ground truth of mid-sized scenes (e.g. rooms, courtyards, etc.). They align the 3D points estimated by SfM to the 3D models to estimate camera pose ground truth.

Only one dataset (EuRoc) uses both a laser scanner for ground truth geometry with a motion capture system for ground truth camera pose. All previously mentioned datasets are from a ground perspective. Also, ground truth camera pose in large indoor scenes is limited, yet accurate camera localization and geometry estimation are important for indoor robot navigation. FEATS mitigates these shortcomings because it provides ground truth camera pose and 3D point locations (enabling 3D point accuracy evaluation) for any scene. We also provide new real world data and camera pose ground truth that focuses on the fundamental motions that are challenging for SfM (e.g. pure rotation).

3. Synthesizing Feature Tracks

FEATS is implemented using the Unity3D game engine [9]. FEATS provides tools for a user to create 3D scenes (from imported meshes and point clouds) and de-

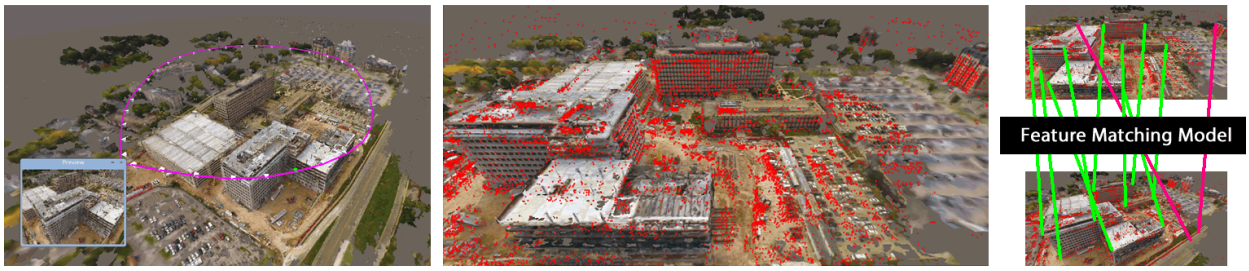


Figure 2: Left: Meshes and point clouds set the 3D scene and cameras (white frustums) set the trajectory (magenta line). Middle: 3D points are back projected onto the cameras as 2D image features (red dots). Right: 2D image features are matched across images using our feature matching model.

fine camera trajectories within those scenes (Section 3.1). With the 3D scene and a trajectory created, 2D feature and matching noise models are used to synthesize feature tracks with real world characteristics (Section 3.2).

3.1. Setting up a 3D Scene

FEATS imports both point clouds and meshes to provide a set of 3D points that can be backprojected as 2D keypoints as the camera moves through the scene. Point clouds generated using SLAM/SfM algorithms are ideal since (1) the 3D points are generated by tracking 2D point locations across images, and (2) the sparsely tracked points encode the feature density of the scene (i.e. as opposed to meshes that are dense surfaces). Meshes provide features for tracking if a point cloud is not provided, but are better suited for occlusion detection (i.e. we do not back project 3D points through walls). Ideally, both a point cloud and aligned mesh are imported together (easy to do using SfM and multi-view stereo [22]), providing both 3D points to track and surfaces for occlusion. Note that incorrect points are okay because they become perfectly accurate ground truth points for future reconstructions.

Trajectories are imported or edited/created by placing cameras. As each camera is placed, an interpolated path is created connecting the cameras sequentially. Cameras can be selected for position and rotation tweaks and a preview window shows the camera's current view. Figure 2 provides a depiction of setting up the 3D world and camera placement.

3.2. Synthesizing Feature Tracks and Ground Truth

FEATS provides options to dictate the density of frames to capture for a camera trajectory. For each frame, feature tracks are synthesized by first finding the 3D points within the viewing area of that camera and backprojecting them to the image plane of the frame according to our 2D feature noise model. Those points are then matched to points in all subsequent frames based on our matching model (Figure 2).

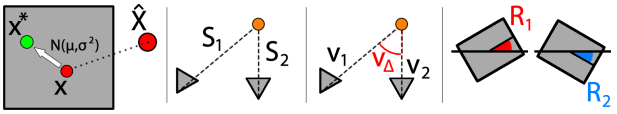


Figure 3: Farthest left: 2D noise being added to backprojected 3D points. Middle left: S_1,S_2 for scale match probability. Middle right: v_1,v_2 for viewing direction match probability. Farthest right: R_1,R_2 for rotation direction match probability.

2D Feature Noise: 3D points are back projected to the image plane of a frame using the projective camera model:

$$x_{ij} = K_j[R_j\ t_j]\hat{X}^i \tag{1}$$

where K_j is the intrinsic camera matrix for image j, $[R_j\ t_j]$ is the extrinsic camera matrix for image j, $\hat{X}^i$ is position of point i in 3D, and x_{ij} is the 2D position of 3D point $\hat{X}^i$ projected onto image j. In the simulator, all $[R_j\ t_j]$ and $\hat{X}^i$ are known. In our experiments, we use the approach from VisualSfM [53] to estimate the focal length of the camera (f) of K as $f = 1.2 * max(W, H)$ and we assume the principal point (cx, cy) of K is $cx = W/2$ and $cy = H/2$. W is frame width and H is frame height.

Noise is then added to each 2D point on the frame (Figure 3):

$$x^* = x + \mathcal{N}(\mu,\ \sigma^2) \tag{2}$$

where x^* is final 2D position of the synthetic feature point, and $\mathcal{N}(\mu,\ \sigma^2)$ is the normal distribution with mean μ and standard deviation σ. We use $\mu = 0$ and $\sigma^2 = 1$ for all results in this paper (except σ^2 varies for Section 5.1).

Matching Model: A probability model is used to dictate whether a feature point matches across two frames. The probability is calculated based on the difference in scale, viewing direction, and rotation (Figure 3). These equations are inspired by the feature matching experiments of Mikolajczyk et al. [37, 36]. The scale match probability

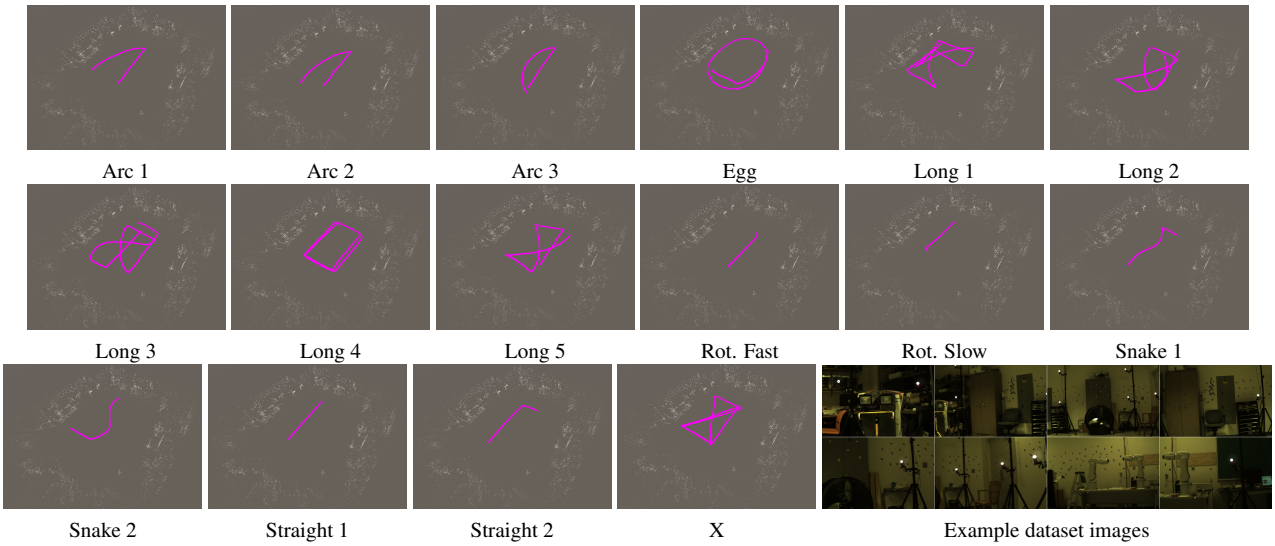


Figure 4: The 16 datasets are depicted. The trajectory of the dataset is depicted as a magenta line. The ORB-SLAM2 map of the motion capture arena is shown as the white point cloud. Example images from the datasets are also shown.

(P_{scale}) is defined as:

$$P_{scale} = P^S_{max} * \exp\left(-\left|\frac{S_\Delta}{\alpha_S}\right|\right)$$
$$S_\Delta = \frac{\max(S_1, S_2)}{\min(S_1, S_2)} - 1 \quad (3)$$

where P^S_{max} is the max probability, α_S is a tuning parameter, and S_1 and S_2 are the distance from the 3D point to each frame's camera center. This model decreases the chances of a match as the difference in scale increases.

The viewing direction match probability (P_{view}) is defined as:

$$P_{view} = P^V_{max} * \exp\left(-\left|\frac{V_\Delta}{\alpha_V}\right|\right)$$
$$V_\Delta = \arccos(v1 \cdot v2) \quad (4)$$

where P^V_{max} is the max probability, α_V is a tuning parameter, and v_1 and v_2 are unit vectors from camera centers of each frame to the 3D point. This model decreases the chances of a match as the difference in viewing direction increases.

The rotation direction match probability (P_{rot}) is defined as:

$$P_{rot} = P^R_{max} - \frac{\alpha_R}{\pi} * R_\Delta$$
$$R_\Delta = \pi - ||R_1 - R_2| - \pi| \quad (5)$$

where P^R_{max} is the max probability, α_R is a tuning parameter, and R_1 and R_2 are the camera roll rotations (i.e. rotation about the look-at vector). This model decreases the chances of a match as the difference in orientation increases. The decay is slower because we found in practice that scale and viewing direction differences effect matching probability more than orientation.

The final probability of a match is:

$$P_{final} = P_{scale} * P_{view} * P_{rot}\,. \quad (6)$$

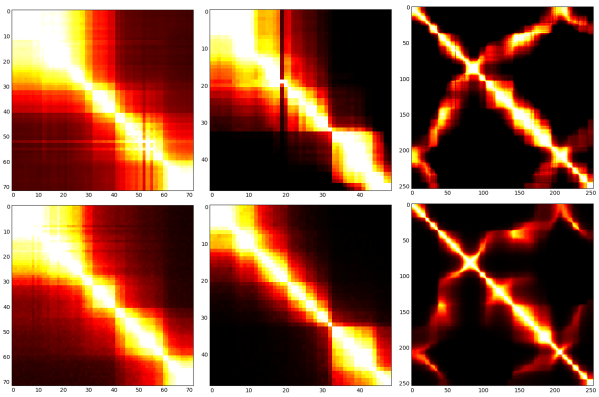
P_{final} is calculated for each feature in each pair of frames and compared against a randomly generated number to decide if it is a match or not. Once all matches for a pair of frames are found, $N_{drop}\%$ of the matches are dropped. Lastly, $N_{bad}\%$ of incorrect matches are also added.

We use $P^S_{max} = 0.9$, $\alpha_S = 2$, $P^V_{max} = 0.9$, $\alpha_V = 6$, $P^R_{max} = 1.0$, $\alpha_R = 0.1$, $N_{drop} = 2\%$, and $N_{bad} = 1\%$ for all results in this paper (except N_{bad} varies for Section 5.1). These values were chosen based on the experimentation in Section 4.2 and the results in [37, 36].

The final output is (1) a feature file for each frame with noisy u,v locations and true X,Y,Z 3D positions of each feature point; and (2) a match file for each frame listing all matches to other frames.

4. Comparing to Real Data

In this section, we present a new dataset (Section 4.1) and use it to show that our matching model is a realistic representation of image matching (Section 4.2); and when SfM packages process our synthesized feature tracks, the pose and geometry outputs are representative of the results that these SfM packages produce on real data (Section 4.3). Comparisons in this paper use three modern SfM pipelines: COLMAP [41], OpenSfM [3], and VisualSfM [53].



Straight2 OpenSfM Rot. Fast COLMAP Long5 VisualSfM

Figure 5: Match percentages (top: real, bottom: synthetic) are shown for the matching steps of COLMAP, VisualSfM, and OpenSfM. All plots are in the same scale (black = 0%, white = 70%). The Pearson correlation r-value between match percentages for "Straight 2" is 0.98 (0.98), "Rotation Fast" is 0.94 (0.92), and "Long 5" is 0.88 (0.84). The r-value in parenthesis is ignoring cells where both match probabilities are below 1%. For all correlation calculations, the diagonal is not used. These r-values indicate strong positive correlation.

4.1. Trajectories with Pose Ground Truth

In total, 17 datasets (image collections with ground truth 6DoF pose) are collected in a motion capture arena (OptiTrack system [4]). The first dataset is strictly used to generate a point cloud of the motion capture arena using ORB-SLAM2 [38]. ORB-SLAM2 is appropriate because it works well for indoor mapping and uses different features (ORB [40]) than those of SfM pipelines (SIFT [34]). We use coherent point drift [46] to align the ORB-SLAM2 trajectory to the ground truth trajectory and apply that transformation to bring the map into the coordinate frame of the motion capture arena.

The other 16 trajectories are for comparison. Each of these trajectories is meant to focus on specific types of motion (some of which are often difficult for SfM). For example, the Arc datasets add progressively more rotation, the straight datasets have sideways and looming translation, and the long datasets close loops. Figure 4 shows each of the 16 trajectories (magenta lines) in the ORB-SLAM2 generated map (white points). The images (colored 752x480 resolution) for these datasets are from a Matrix Vision mvBluefox-200wc camera [2]. Figure 4 shows example images from the datasets. See the supplementary material for additional information.

	COLMAP	OpenSfM	VisualSfM
Arc 1	0.84 (0.82)	0.91 (0.89)	0.78 (0.78)
Arc 2	0.87 (0.85)	0.92 (0.90)	0.80 (0.80)
Arc 3	0.86 (0.83)	0.96 (0.95)	0.93 (0.90)
Egg	0.88 (0.82)	0.91 (0.88)	0.84 (0.81)
Long 1	0.87 (0.82)	0.91 (0.89)	0.84 (0.78)
Long 2	0.90 (0.88)	0.91 (0.90)	0.86 (0.84)
Long 3	0.90 (0.87)	0.93 (0.91)	0.84 (0.84)
Long 4	0.90 (0.88)	0.93 (0.92)	0.88 (0.86)
Long 5	0.89 (0.86)	0.91 (0.89)	0.88 (0.84)
Pure rotation fast	0.94 (0.92)	0.94 (0.92)	0.93 (0.91)
Pure rotation slow	0.89 (0.87)	0.91 (0.88)	0.89 (0.87)
Snake 1	0.91 (0.91)	0.96 (0.96)	0.83 (0.83)
Snake 2	0.83 (0.83)	0.92 (0.92)	0.74 (0.74)
Straight 1	0.85 (0.85)	0.95 (0.95)	0.74 (0.74)
Straight 2	0.94 (0.94)	0.98 (0.98)	0.82 (0.82)
Trajectory X	0.90 (0.90)	0.95 (0.95)	0.88 (0.88)

Table 1: Pearson correlation r-values for the match percentages of real and synthetic data for the matching step of COLMAP, OpenSfM, and VisualSfM. The value in parenthesis is the r-value ignoring cells of the matrix where both probabilities are below 1% (i.e. close to 0%). All values are above 0.74, indicating a strong positive correlation.

4.2. Verifying the Match Model

FEATS matching is highly predictive of matching on real images: Figure 5 shows comparisons (top: real, bottom: simulated) of the percentage of matches between image pairs for the matching step of COLMAP, OpenSfM, and VisualSfM. In each plot, the x and y axis are sequential frames. For example, reading across row 1 shows the percentage of matches for image 1 compared to each other image. Continuing with this example, the match percentage for row 1, column 10 (i.e. image 1 matching to image 10) is the number of feature matches between image 1 and 10 divided by the number of features in image one. On the other hand, the match percentage for row 10, column 1 is the number of feature matches between image 10 and 1 divided by the number of features in image 10. The scale of all plots is the same (black = 0% and white = 70%).

The match percentages accurately reflect effects due to trajectory. For example, the "Straight 2" trajectory has very little rotation and mostly looming motion. Thus, we see decreasing match percentages because the scale difference increases (i.e. the chances of matches decreases as the scale difference increases). Similarly, "Rotation Fast" has shifting translation followed by significant out-of-plane rotation, causing the viewing angle difference to increase drastically and reduce the matching percentage around frame 30. For longer datasets with significant translation and rotation (i.e. "Long 5"), we see qualitatively that the synthetic match percentages represent the real match percentages well. See supplementary material for all match percentage plots.

Table 1 provides the Pearson correlation r-values between each real and simulated match percentage matrix. The diagonal is ignored because each diagonal cell is an image matching with itself. The values in parenthesis are

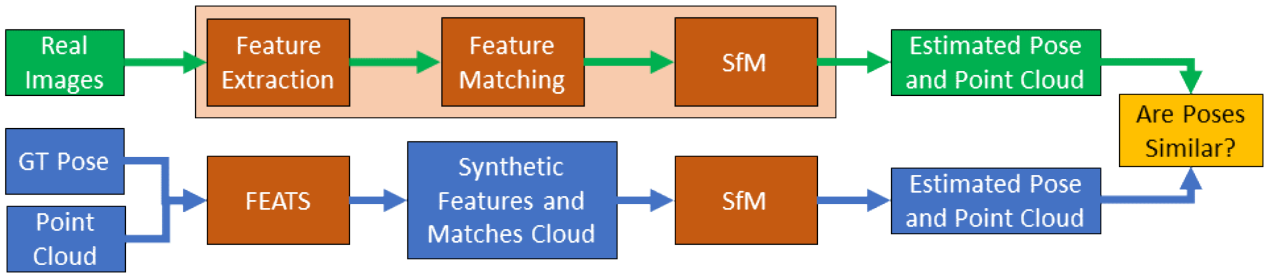


Figure 6: We show that synthetic tracks are predictive of results from real data using the following experiment. There are two flows from left to right: real images (green boxes and arrows) and simulated tracks (blue boxes and arrows). Real images are input into SfM and pose and geometry are output. The ground truth (GT) pose for the real images and motion capture arena point cloud are input to FEATS. The GT Pose is used by the simulator to generate a camera trajectory through the point cloud. Simulated tracks are output and run with SfM to generate pose and geometry estimates. The output pose from the synthetic and real data are compared to show that the simulated tracks represent real data.

the Pearson correlation r-values with the "zero" probabilities excluded. Specifically, if, for a given cell of the matrix, both the real and simulated match percentages are below 1%, then that value is excluded. For all trajectories for all three SfM methods, the r-values are 0.74 or greater. This indicates a strong positive correlation between the real and synthetic match percentages (values above 0.5 are typically considered a strong correlation [15]), providing evidence that the FEATS matching model represents matching of real images by COLMAP, OpenSfM, and VisualSfM.

We also calculate the average L2 distance between each real and simulated match percentage matrix. The total average L2 distance is less than 0.10% for all algorithms and all trajectories. These numbers show that there is no significant scaling or translation between the match percentages, and provide additional evidence that the match model accurately represents real matching for COLMAP, OpenSfM, and VisualSfM. See supplementary material for all values.

4.3. Verifying the Synthetic Tracks

Figure 6 outlines our experimental setup. There are two flows from left to right: one for real data (shown with green boxes and lines) and one for synthetic data (shown with blue boxes and lines). For clarification, ground truth pose (GT Pose) is real data, but is a blue box because it is used to set the trajectories in the simulator 3D scene (Section 3.1).

For the real data, images are input into the SfM pipeline and the output is pose and geometry. For the synthetic data, the ORB-SLAM2 point cloud of the motion capture arena and GT Pose are input into the simulator for each dataset. The simulator uses the GT Pose to define the camera trajectory. FEATS uses the feature noise and match model to generate synthetic feature tracks that are input to SfM. The output is pose and geometry. We align the real and synthetic pose estimates to the GT pose using Horn's method [31].

This process is repeated for all 16 trajectories and using three state of the art SfM pipelines: COLMAP [41], OpenSfM [3], and VisualSfM [53]. We manually inspect each set of trajectories and tabulate in Table 2 whether the reconstructions are successful or not ("F" denotes failure). Examples of the aligned trajectories are shown in Figure 7.

	COLMAP			OpenSfM			VisualSfM		
Arc 1		/			/			/	
Arc 2		/			/	F		/	
Arc 3		/			/		F	/	F
Egg	F	/	F		/		F	/	F
Long 1		/		F	/	F	F	/	F
Long 2		/			/			/	
Long 3		/			/			/	
Long 4		/			/			/	
Long 5		/			/			/	
Rotation Fast		/		F	/	F		/	
Rotation Slow		/		F	/	F		/	
Snake 1		/			/			/	
Snake 2		/			/	F		/	
Straight 1		/			/		F	/	F
Straight 2		/			/			/	
Trajectory X		/			/			/	

Table 2: Blank entries indicate that the reconstruction is correct. The notation is (real/synthetic). "F" indicates failure. There are 8 failures on real data and the synthetic data predicts 8 of them (recall of 100%). The synthetic data incorrectly predicts 2 failures that do not occur (precision of 80%). The overall accuracy is 96% (46/48). This is evidence that the synthetic data represents the real data well and helps predict failure reconstructions.

FEATS successfully predicts success and failure: From Table 2, we see that the three SfM algorithms fail 8 times in total on the real data and all of those failures are predicted by the synthetic data (recall of 100%). The simulated data predicts failure 10 times in total (precision is 80%). For classifying success, recall is 100% and precision is 95%. **Overall, the accuracy is 96% (46/48).** These results are evidence that the synthetic data is an effective representation of the real data. Moreover, these results pro-

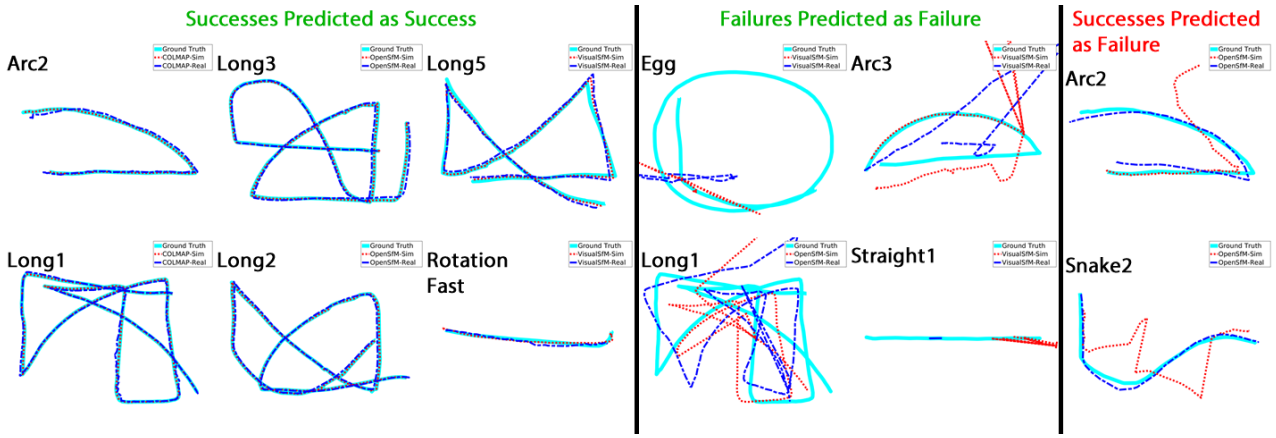


Figure 7: Trajectories for synthetic (red dotted) and real (blue dashed) data are aligned to ground truth (cyan solid). The left section shows successful reconstructions that the synthetic data also predicts as successful (column 1 for COLMAP, column 2 for OpenSfM, column 3 for VisualSfM). The middle section is failure reconstructions that the synthetic data also predicts as failures. The right section is successful reconstructions that are predicted as failures.

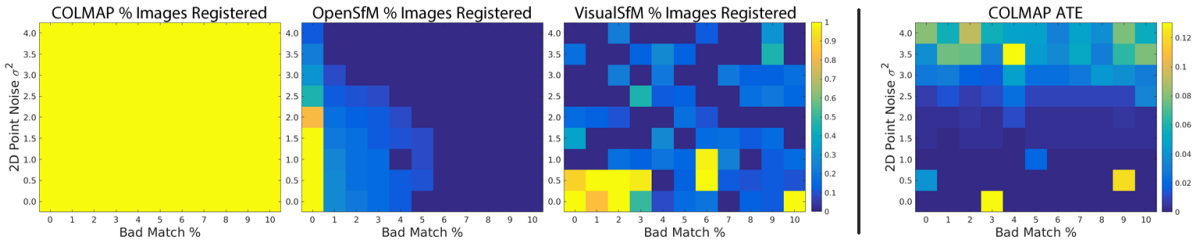


Figure 8: The left three plots show the percent of localized images as 2D gaussian noise σ^2 and bad match percent vary (yellow = 100%, dark blue = 0%). The right plot shows that absolute trajectory error [49] generally increases as noise increases for COLMAP.

vide evidence that FEATS can reliably predict failure reconstructions, which can be used to test and adjust planned data captures before time and effort is spent collecting the data.

5. SfM Evaluations Enabled by FEATS

In this section, we demonstrate two new methods to evaluate SfM that are enabled by FEATS: (1) comparing SfM as 2D feature noise varies and (2) calculating 3D point error.

5.1. Robustness to Feature Noise and Bad Matches

Using FEATS, we generate 99 trajectories of Arc1 while varying the percent of bad matches and σ^2 in Equation 2. We vary the percent of bad matches between 0% and 10% by increments of 1%. We vary σ^2 between 0 and 4 by increments of 0.5. We process these 99 trajectories with COLMAP, OpenSfM, and VisualSfM. The percent of localized images for each algorithm is shown in Figure 8.

COLMAP is robust to both types of noise, registering all images. OpenSfM is sensitive to bad matches. On the other hand, VisualSfM is less sensitive to bad matches, but more sensitive to 2D feature noise. Comparing COLMAP to OpenSfM provides evidence that COLMAP's scene graph augmentation and new triangulation approach [41] (implementations that differ for OpenSfM) are effective in overcoming 2D noise and bad matches. For VisualSfM, the noise causes inconsistent results, which OpenSfM and COLMAP may not exhibit because COLMAP and OpenSfM have additional outlier filtering and retriangulation methods used during bundle adjustment. In all cases, failed reconstructions are marked as 0% and incorrectly registered images are not counted.

Since COLMAP registers all images for all trajectories, we calculate the absolute trajectory error (ATE) [49] in the right plot of Figure 8. It is interesting to see how the ATE increases significantly as the 2D noise approaches $\sigma^2 = 4$.

5.2. Calculating 3D Point Error

It is challenging to evaluate the accuracy of a 3D reconstructed point cloud quantitatively. Instead, qualitative measures are used (i.e. does the point cloud look "good" and "clean"). Even when ground truth geometry is provided

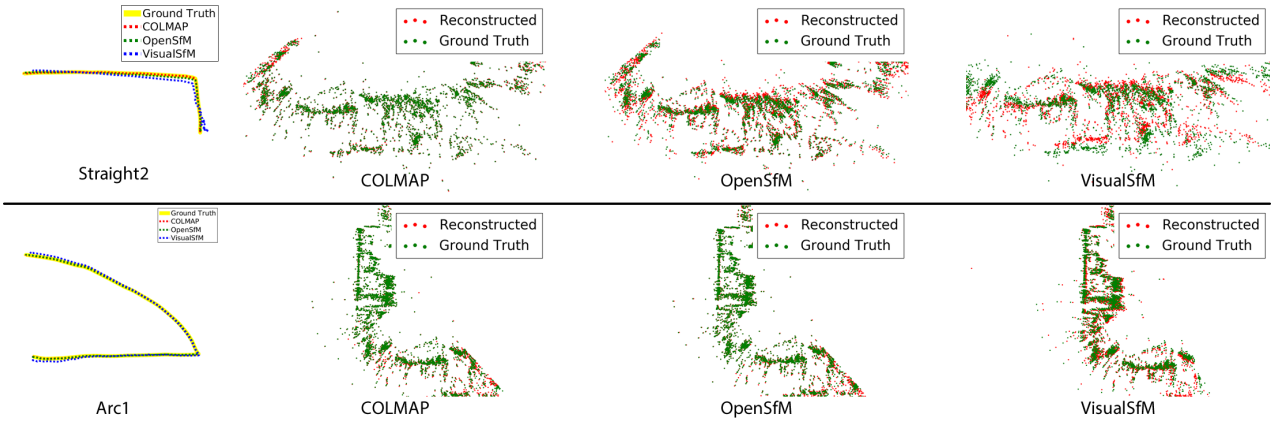


Figure 9: The ground truth point cloud (i.e. map points used by FEATS) is shown in green. The reconstructed point cloud (i.e. synthetic feature tracks processed by SfM) is shown in red. In the bottom two figures, the trajectories are plotted, showing that pose error is low; however, the 3D point errors are different for each SfM (more red indicates more error).

by a CAD model or laser scanner, it is difficult to find the corresponding point on the ground truth mesh for each 3D reconstructed point. The typical approach is to fit the point cloud to the mesh and find the shortest distance between each point and the mesh. This is not ideal for quantitative evaluation of the quality of the reconstructed point cloud.

With FEATS, for each reconstructed 3D point, the corresponding ground truth 3D point coordinate is known. This makes it easy to calculate the error between these points. We calculate the average L2 point error for each reconstructed point cloud and ground truth point cloud using the following equation:

$$\frac{1}{N}\sum_{i=1}^{N}||X_i - X_i^*||_2 \tag{7}$$

where N is the number of points in the reconstruction, X_i is 3D point i from the reconstruction, and X_i^* is the ground truth 3D point corresponding to X_i. The resulting average point errors (in millimeters) are provided in Table 3. Failed reconstructions are not included and egregious (10 meters or more) outlier points are also not included in the calculation.

Table 3 shows that COLMAP and OpenSfM provide the most accurate point clouds. This is evidence that the retriangulation and bundle adjustment additions in these approaches [41] are effective at improving point cloud accuracy over previous methods (e.g. VisualSfM). Figure 9 provides examples of aligned trajectories and points for all three SfM approaches for Arc1 and Straight2. These plots show that the camera localizations are accurate for all three algorithms, yet the point error is noticeably different (i.e. tens vs hundreds of millimeter errors according to Table 3). This indicates that accurate pose estimates do not guarantee accurate 3D point locations, reinforcing the need for new quantitative metrics for measuring 3D point error.

	COLMAP	OpenSfM	VisualSfM
Arc 1	**21.5**	26.0	84.7
Arc 2	**23.0**	-	110.7
Arc 3	**16.8**	22.2	-
Egg	-	**20.2**	-
Long 1	**8.3**	-	-
Long 2	**9.0**	12.9	461.2
Long 3	10.7	**10.6**	138.8
Long 4	9.3	**7.3**	100.3
Long 5	**5.1**	7.3	70.1
Pure rotation fast	**33.2**	-	208.5
Pure rotation slow	**10.8**	-	212.1
Snake 1	**18.6**	271.7	155.2
Snake 2	**31.1**	-	171.9
Straight 1	**20.6**	171.0	-
Straight 2	**27.2**	118.1	246.8
Trajectory X	**7.0**	11.7	109.1

Table 3: Average 3D point error (in millimeters) is calculated. The lowest errors are bold. Failure reconstructions are denoted by "-". Points more than 10 meters from their ground truth position are not included.

6. Conclusions

We present FEATS, a simulation environment that models feature and matching noise to generate feature tracks from camera trajectories in a virtual 3D scene. We show the synthetic tracks are representative of real world data by comparing the percentage of matches between image pairs of real and synthetic data and by using the simulated data to predict reconstruction successes and failures. We use synthetic data to show that (1) COLMAP is quite robust to 2D feature noise and bad matches; and (2) accurate camera localizations do not guarantee accurate point clouds, which reinforces the need for ground truth 3D point error.

Acknowledgement. This work is supported by NSF Grant CMMI-1446765 and the DoD National Defense Science and Engineering Graduate Fellowship (NDSEG). Thank you also to Reconstruct for computational resources that enabled this research.

References

[1] Drone deploy. https://www.dronedeploy.com/.

[2] Matrix vision mvbluefox-200wc. https://www.matrix-vision.com/usb2.0-industrial-camera-mvbluefox.html.

[3] Opensfm. https://github.com/mapillary/opensfm.

[4] Optitrack - motion capture systems. http://optitrack.com/.

[5] Pix4d. https://pix4d.com/.

[6] Rawseeds project. http://www.rawseeds.org/home/.

[7] Reconstruct. https://www.reconstructinc.com/.

[8] Bigsfm: Reconstructing the world from internet photos. http://www.cs.cornell.edu/projects/bigsfm/, October 2016.

[9] Unity - game engine. https://unity3d.com/, October 2016.

[10] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *EEE 12th International Conference on Computer Vision*, pages 72–79, Sept 2009.

[11] P. W. Battaglia, J. B. Hamrick, and J. B. Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332, 2013.

[12] J.-L. Blanco, F.-A. Moreno, and J. González. A collection of outdoor robotic datasets with centimeter-accuracy ground truth. *Autonomous Robots*, 27(4):327–351, November 2009.

[13] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 2016.

[14] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, Oct. 2012.

[15] J. Cohen. *Statistical Power Analysis for the Behavioral Sciences*. 1988.

[16] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. SfM with MRFs: Discrete-continuous optimization for large-scale structure from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 35(12):2841–2853, December 2013.

[17] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, 2017.

[18] C. Eschmann, C.-M. Kuo, C.-H. Kuo, and C. Boller. Unmanned aircraft systems for remote building inspection and monitoring. 2012.

[19] J. Fernandez-Galarreta, N. Kerle, and M. Gerke. Uav-based urban structural damage assessment using object-based image analysis and semantic reasoning. 15:1087–1101, 06 2015.

[20] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, and P. Hanrahan. Example-based synthesis of 3d object arrangements. In *ACM SIGGRAPH Asia 2012 papers*, SIGGRAPH Asia '12, 2012.

[21] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys. Building rome on a cloudless day. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, 2010.

[22] S. Fuhrmann, F. Langguth, N. Moehrle, M. Waechter, and M. Goesele. Mve – an image-based reconstruction environment. *Computer and Graphics*, 2015.

[23] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. Virtual worlds as proxy for multi-object tracking analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[24] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3354–3361, June 2012.

[25] Y. Ham, K. K. Han, J. J. Lin, and M. Golparvar-Fard. Visual monitoring of civil infrastructure systems via camera-equipped unmanned aerial vehicles (uavs): a review of related works. *Visualization in Engineering*, 2016.

[26] K. Han, J. Degol, and M. Golparvar-Fard. Geometry- and appearance-based reasoning of construction progress monitoring. *Journal of Construction Engineering and Management*, 2018.

[27] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla. Understanding real world indoor scenes with synthetic data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[28] A. Handa, V. Pătrăucean, S. Stent, and R. Cipolla. Scenenet: An annotated model generator for indoor scene understanding. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5737–5743, May 2016.

[29] A. Handa, T. Whelan, J. McDonald, and A. J. Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In *ICRA*, 2014.

[30] H. Hattori, V. N. Boddeti, K. Kitani, and T. Kanade. Learning scene-specific pedestrian detectors without real data. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3819–3827, June 2015.

[31] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A*, 1987.

[32] B. Kaneva, A. Torralba, and W. T. Freeman. Evaluating image feaures using a photorealistic virtual world. In *IEEE International Conference on Computer Vision*, 2011.

[33] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 2017.

[34] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004.

[35] J. Marín, D. Vázquez, D. Gerónimo, and A. M. López. Learning appearance in virtual scenarios for pedestrian detection. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 137–144, June 2010.

[36] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005.

[37] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *Int. J. Comput. Vision*, 2005.

[38] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[39] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[40] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, 2011.

[41] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[42] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[43] S. Shah, D. Dey, C. Lovett, and A. Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017.

[44] J. Shotton, A. Fitzgibbon, , A. Blake, A. Kipman, M. Finocchio, R. Moore, and T. Sharp. Real-time human pose recognition in parts from a single depth image. In *CVPR*, 2011.

[45] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Trans. Graph.*, 2006.

[46] X. Song and A. Myronenko. Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[47] M. Stark, M. Goesele, and B. Schiele. Back to the future: Learning shape models from 3d cad data. In *Proc. BMVC*, pages 106.1–11, 2010.

[48] C. Strecha, T. Pylvänäinen, and P. Fua. Dynamic and scalable large scale image reconstruction. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2010.

[49] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580, Oct 2012.

[50] G. R. Taylor, A. J. Chosak, and P. C. Brewer. Ovvv: Using virtual worlds to design and evaluate surveillance systems. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[51] T. Vaudrey, C. Rabe, R. Klette, and J. Milburn. Differences between stereo and motion behaviour on synthetic and real-world stereo sequences. In *2008 23rd International Conference Image and Vision Computing New Zealand*, pages 1–6, Nov 2008.

[52] D. Vázquez, A. M. López, J. Marín, D. Ponsa, and D. Gerónimo. Virtual and real world adaptation for pedestrian detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):797–809, April 2014.

[53] C. Wu. Towards linear-time incremental structure from motion. In *2013 International Conference on 3D Vision - 3DV 2013*, pages 127–134, June 2013.

[54] J. Xu, S. Ramos, D. Vázquez, and A. M. López. Hierarchical adaptive structural svm for domain adaptation. *International Journal of Computer Vision*, 2016.