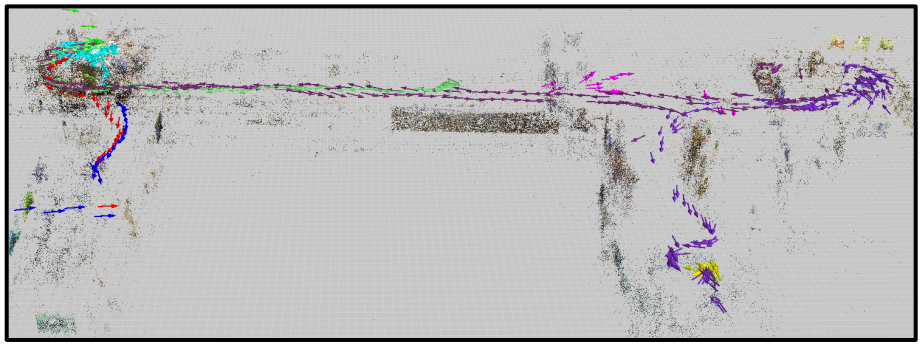
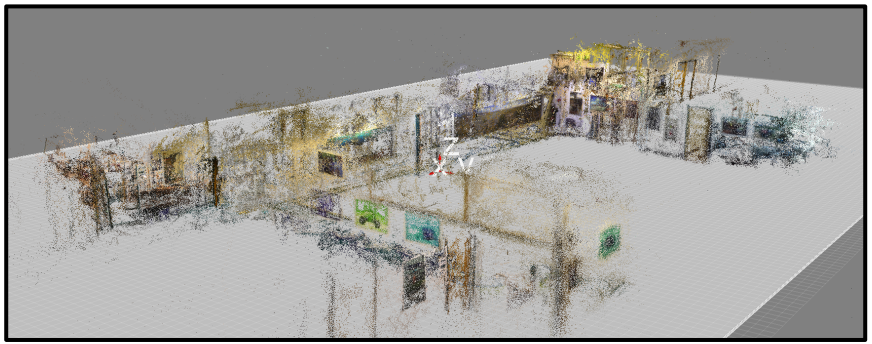
Fast SFM-Based Localization of Temporal Sequences and Ground-Plane Hypothesis Consensus





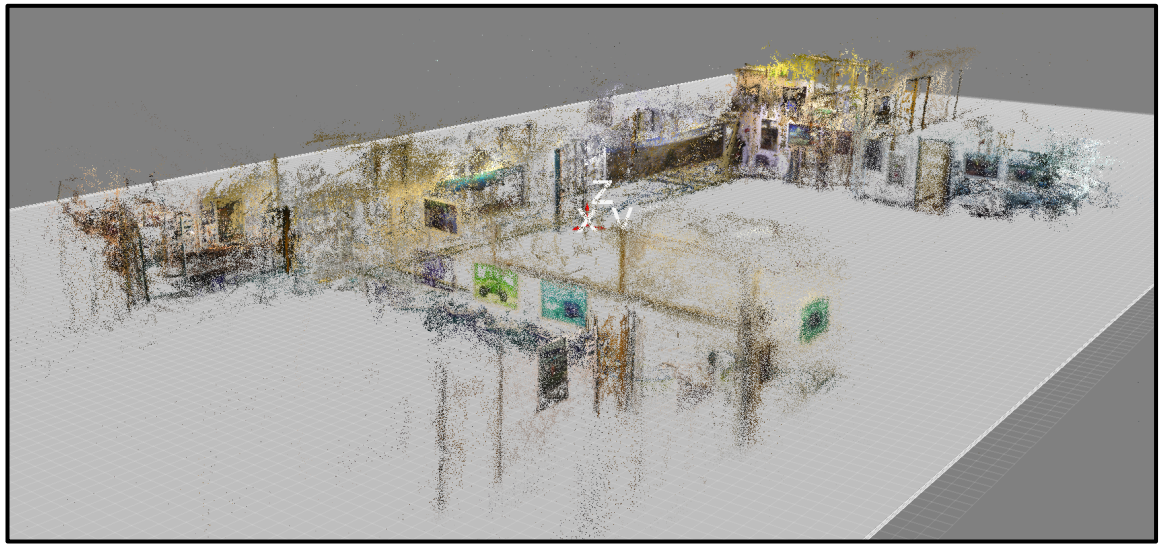
Nick Rhinehart

May 2015

Carnegie Mellon University

Goals

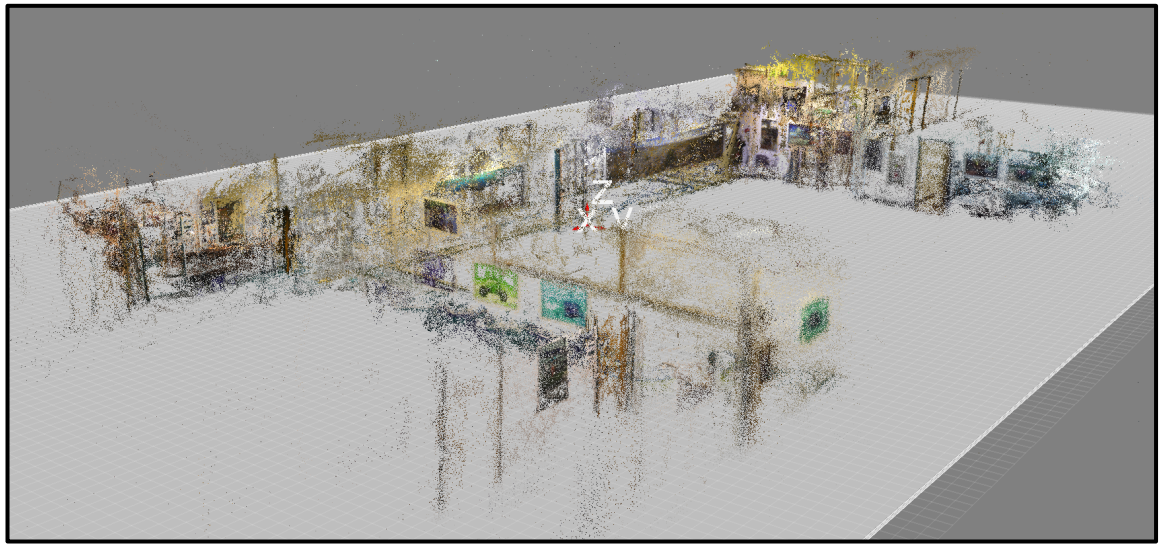
- Estimate ground plane from SFM cameras and image-based evidence
- Quickly estimate pose of new camera in temporal sequence in already seen environment



SFM from [3], Dense points from PMVS [1], Estimate of global plane

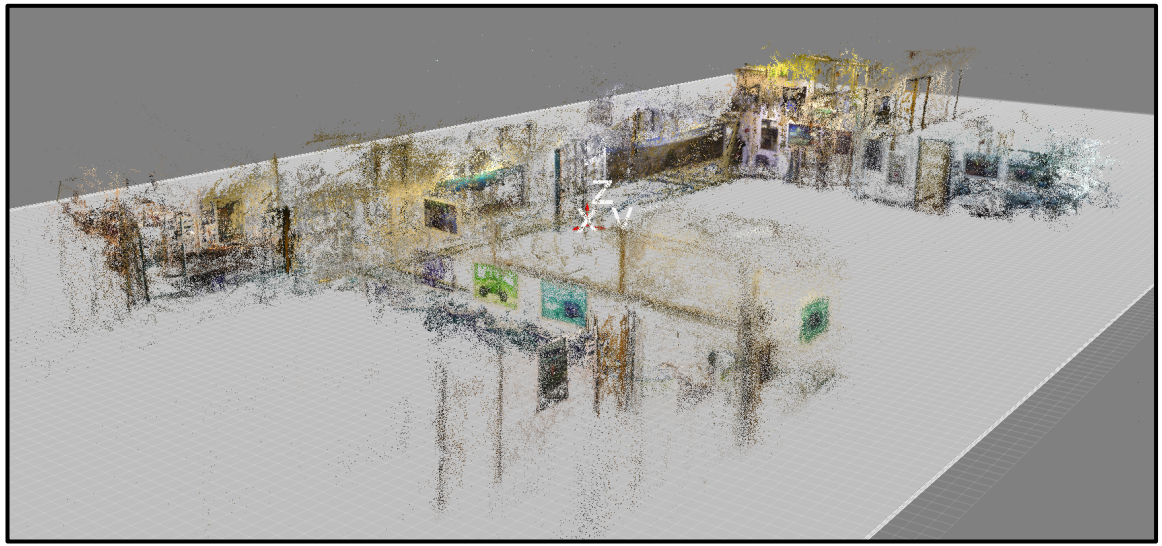
Goal 1: Ground plane estimation

- Points from SFM can be noisy, can't necessarily fit plane simply to ground plane
- Vanishing point methods can work but aren't globally consistent (single image)



Goal 1: Ground plane estimation

- Points from SFM can be noisy, can't necessarily fit plane simply to ground plane
- Vanishing point methods can work but aren't globally consistent (single image)
- **Use vanishing points & image evidence to estimate ground plane for each image, consolidate all estimates**



Ground plane from vanishing points

- Robust 3-direction vanishing point estimation of Manhattan world
- Also has per-pixel surface estimates

Images from applying method of [2]

[2] Hedau, Varsha, Derek Hoiem, and David Forsyth. "Recovering the spatial layout of cluttered rooms." *Computer vision, 2009 IEEE 12th international conference on*. IEEE, 2009.

Ground plane from vanishing points

- Robust 3-direction vanishing point estimation of Manhattan world
- Also has per-pixel surface estimates
- Can recover ground-plane normal estimate by using calibration matrix estimates from SFM

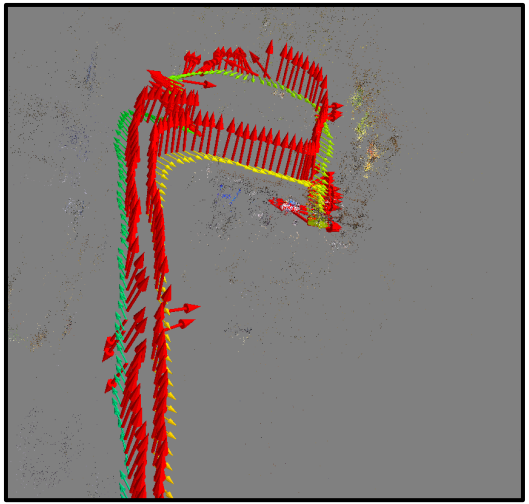
$$n_{ground} = K^{-1} v p_{vertical}$$

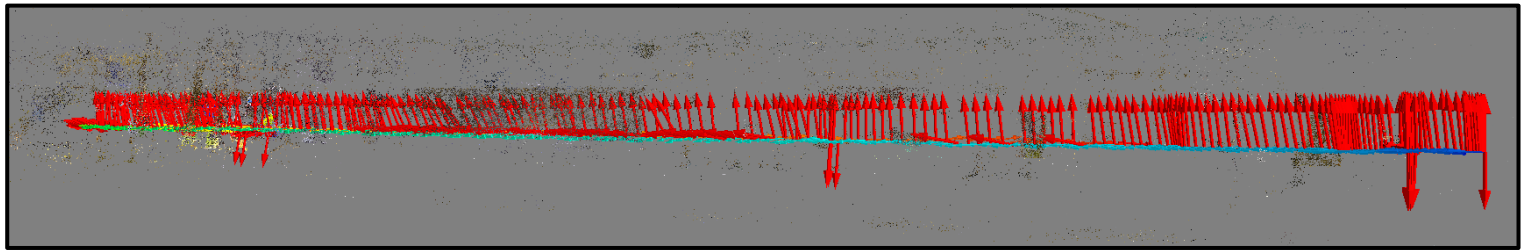
Images from applying method of [2]

[2] Hedau, Varsha, Derek Hoiem, and David Forsyth. "Recovering the spatial layout of cluttered rooms." *Computer vision, 2009 IEEE 12th international conference on*. IEEE, 2009.

Ground plane normals

Most normals are reasonable, some are completely wrong





Plane offset estimation

- For most cameras, reasonably good estimate of plane normal. What about plane offset?

$$\pi = [\mathbf{n}, d]$$

Plane offset estimation

- For most cameras, reasonably good estimate of plane normal. What about plane offset?

$$\pi = [\mathbf{n}, d]$$

Apply per-pixel ground estimate as filter to SFM keypoints

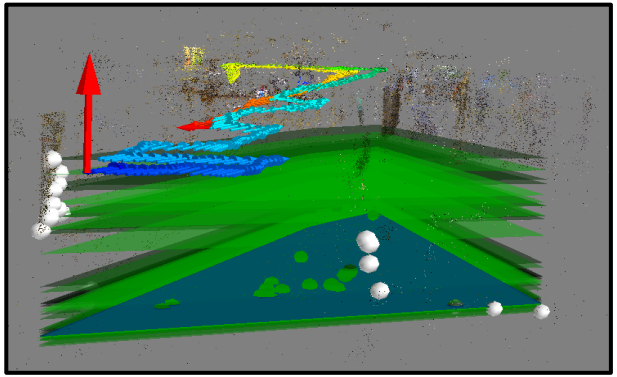
Plane offset estimation

- For most cameras, reasonably good estimate of plane normal. What about plane offset?

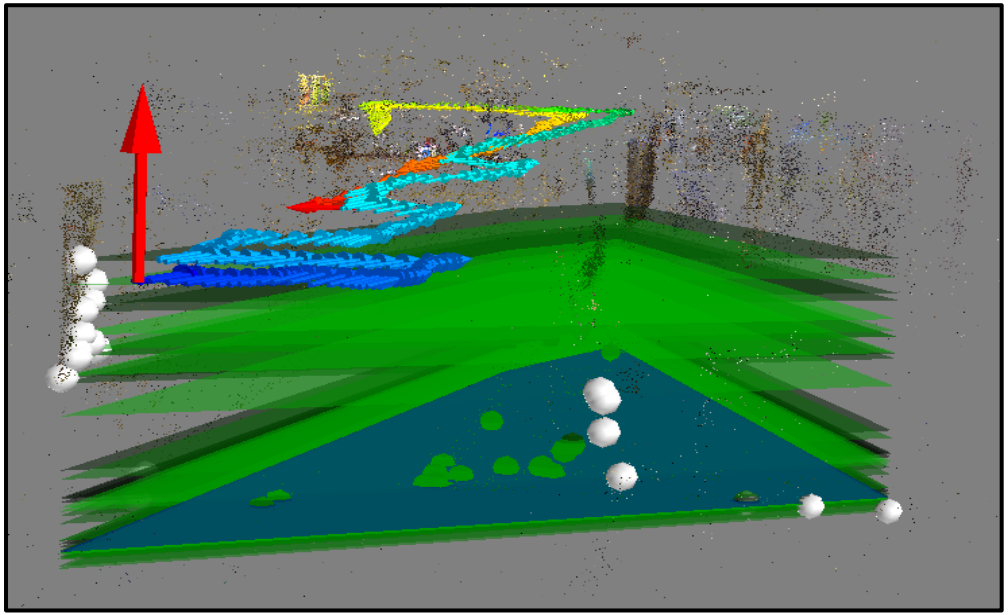
$$\pi = [\mathbf{n}, d]$$

Apply per-pixel ground estimate as filter to SFM keypoints

Slide plane along normal, measure consensus with “ground” keypoints



Plane offset estimation



Maximum consensus plane and candidate planes formed from estimated normal in consensus with estimated ground keypoints

Plot for all ground planes

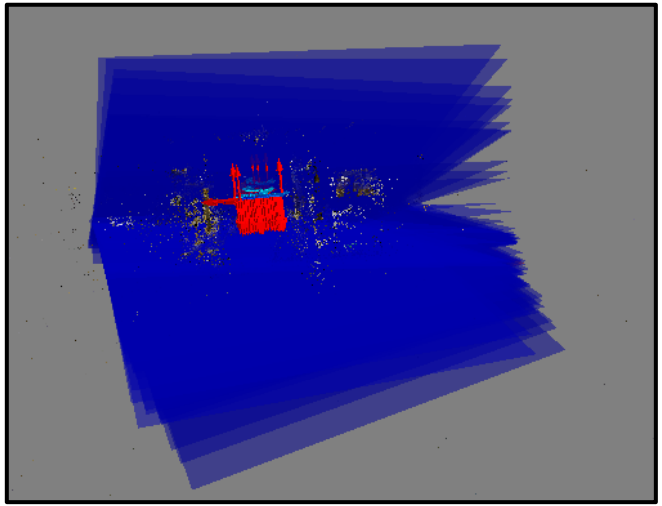
- looks messy, many wrong or offset

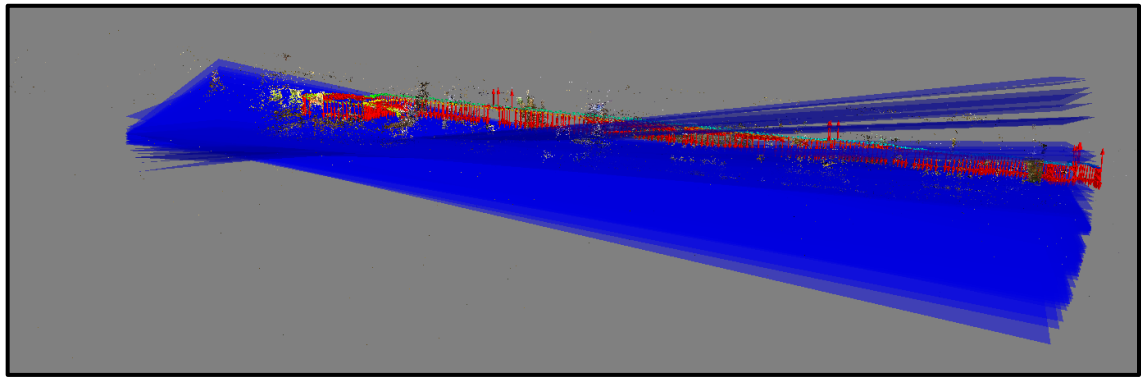
http://www.cs.cmu.edu/~nrhineha/.hidden/videos/planes_all.mp4
(https://www.dropbox.com/s/jdffy11b4iudvdq/planes_all.mp4?dl=0)

Finding global best plane

- have set of planes $\Pi = \{\pi_1, \ldots, \pi_N\}$
- RANSAC across estimated planes with consensus test on angular offset and distance offset, then LSQ fit on plane with maximal consensus

$$C(\pi_i, \pi_j; \alpha, \beta) = \mathbf{1}\left(\cos^{-1}\left(\frac{\mathbf{n_i}^T \mathbf{n_j}}{\|\mathbf{n_i}\| \cdot \|\mathbf{n_j}\|}\right) \leq \alpha\right) \cdot \mathbf{1}\left(|d_i - d_j| \leq \beta\right)$$

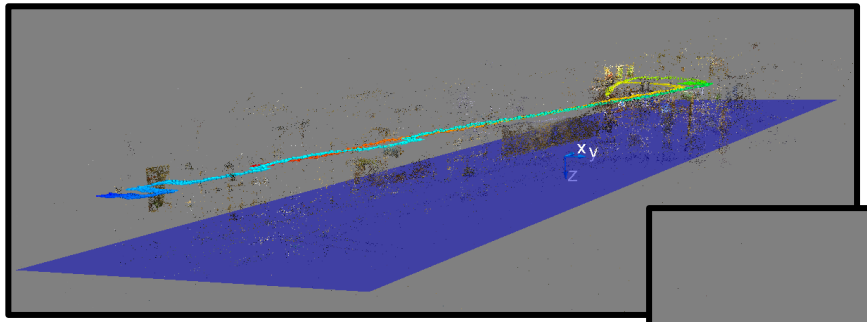




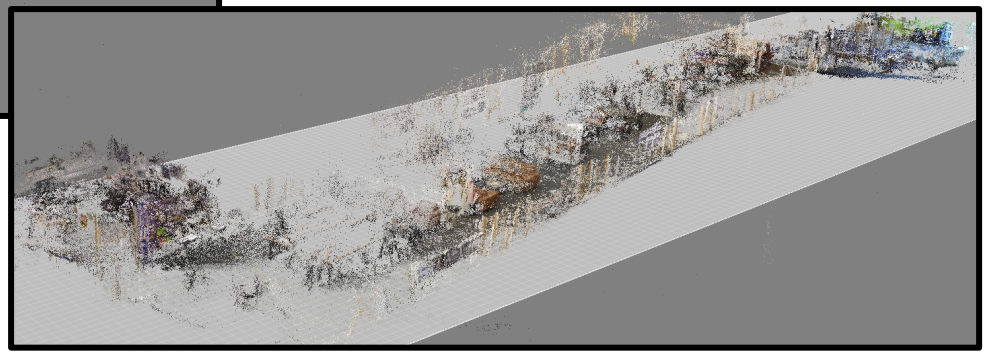
Consensus set of maximal consensus plane (largest consensus set)

Finding global best plane

Global best plane estimate:



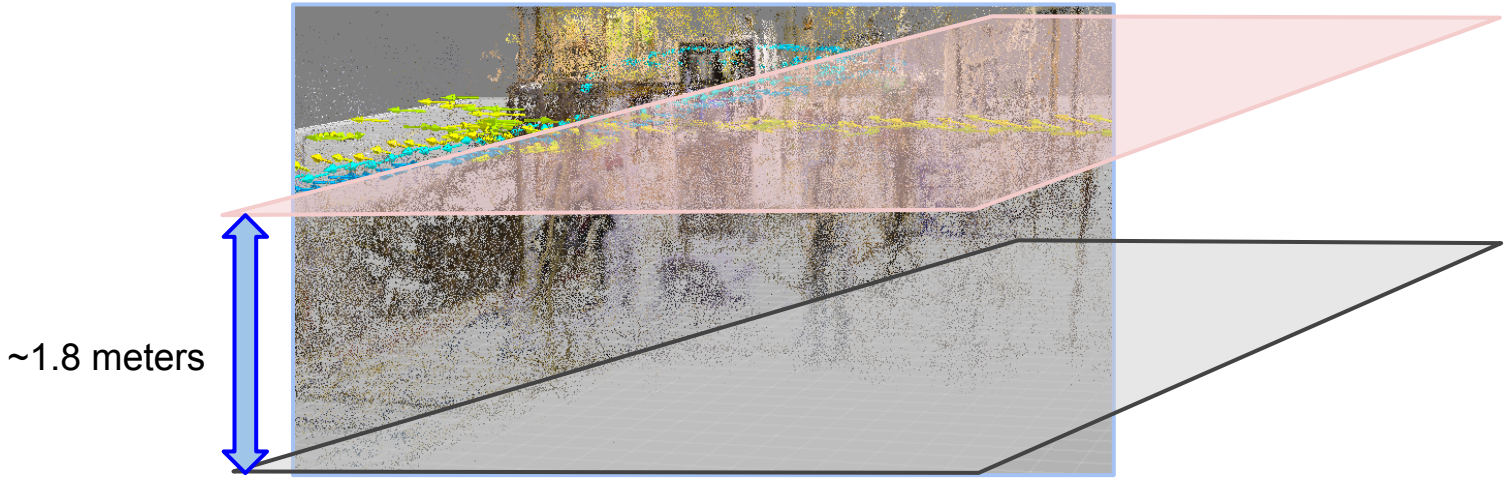
NSH 4th (sparse keypoints)



Smith 2nd (dense keypoints)

Scale bonus

- Images came from single user (egocentric camera).
- *Apply knowledge of user's height*, distance between plane fit to camera (using normal from final ground plane)



Visualization

http://www.cs.cmu.edu/~nrhineha/.hidden/videos/nsh4rotate_vis2.mp4
(https://www.dropbox.com/s/tnvtp3retuumkw2/nsh4rotate_vis2.mp4?dl=0)

http://www.cs.cmu.edu/~nrhineha/.hidden/videos/smith2rotate_vis.mp4
(https://www.dropbox.com/s/w1qzln37zypc54h/smith2rotate_vis.mp4?dl=0)

Goal 2: Fast estimation of temporal sequence in mapped environment

Fast estimation of camera pose in temporal sequence

- Can localize a camera in SFM model by performing all-pairs matching (slow, expensive) + partial bundle adjustment (adjusting only parameters of new camera)

- In large environments, prior on location is very useful to constrain matching

Using prior on location in temporal sequence

Algorithm 1: Temporal Sequence Localization

Input: Sequence of video frames F, SFM model M
Output: Sequence of camera positions P in global reference frame

Using prior on location in temporal sequence

0. Build KD-tree of positions of model cameras

Algorithm 1: Temporal Sequence Localization

Input: Sequence of video frames F, SFM model M

Output: Sequence of camera positions P in global reference frame

```
1 cur_pos = None ;
2 tree = KDTree(M.get_camera_positions());
```

Using prior on location in temporal sequence

0. Build KD-tree of positions of model cameras
1. First camera in sequence matched against all frames

Algorithm 1: Temporal Sequence Localization

Input: Sequence of video frames F, SFM model M
Output: Sequence of camera positions P in global reference frame

```
1  cur_pos = None ;
2  tree = KDTree(M.get_camera_positions());
3  radii = InitializeRadii(); /* create list of a few small radii    */
4  P = [];
5  for f ∈ F do
6      for radius ∈ radii do
7          if cur_pos ≠ None then
8              nearby_cameras = tree.query_radius(cur_pos, radius);
9              edges = M.match_pairwise(f, nearby_cameras.get_frames());
10         else
11             edges = M.match_pairwise(f, M.get_camera_frames());
```

Using prior on location in temporal sequence

0. Build KD-tree of positions of model cameras
1. First camera in sequence matched against all frames
2. Partial bundle adjustment to estimate camera intrinsics & extrinsics

Algorithm 1: Temporal Sequence Localization

Input: Sequence of video frames F, SFM model M
Output: Sequence of camera positions P in global reference frame

```
1  cur_pos = None ;
2  tree = KDTree(M.get_camera_positions());
3  radii = InitializeRadii(); /* create list of a few small radii    */
4  P = [];
5  for f ∈ F do
6      for radius ∈ radii do
7          if cur_pos ≠ None then
8              nearby_cameras = tree.query_radius(cur_pos, radius);
9              edges = M.match_pairwise(f, nearby_cameras.get_frames());
10         else
11             edges = M.match_pairwise(f, M.get_camera_frames());
12         end
13         M.partial_bundle_adjustment(edges);
14         cur_pos = M.get_localized_position(f)
```

Using prior on location in temporal sequence

0. Build KD-tree of positions of model cameras
1. First camera in sequence matched against all frames
2. Partial bundle adjustment to estimate camera intrinsics & extrinsics
3. Future frames matched only to nearby cameras

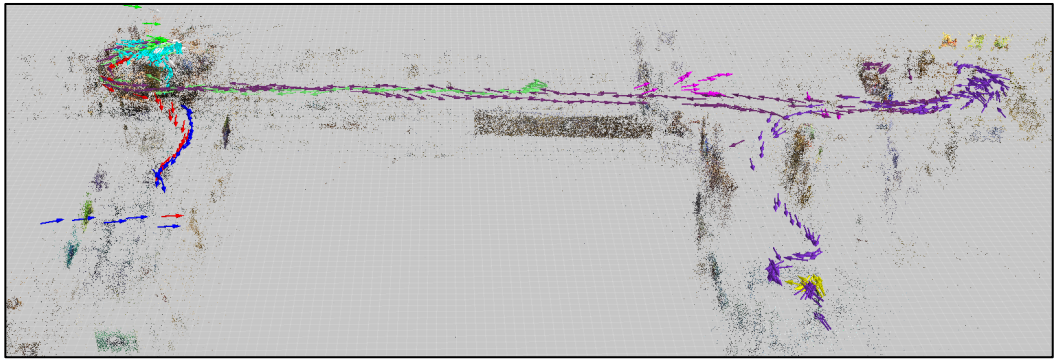
Algorithm 1: Temporal Sequence Localization

Input: Sequence of video frames F, SFM model M
Output: Sequence of camera positions P in global reference frame

```
1  cur_pos = None ;
2  tree = KDTree(M.get_camera_positions());
3  radii = InitializeRadii(); /* create list of a few small radii    */
4  P = [];
5  for f ∈ F do
6      for radius ∈ radii do
7          if cur_pos ≠ None then
8              nearby_cameras = tree.query_radius(cur_pos, radius);
9              edges = M.match_pairwise(f, nearby_cameras.get_frames());
10         else
11             edges = M.match_pairwise(f, M.get_camera_frames());
12         end
13         M.partial_bundle_adjustment(edges);
14         cur_pos = M.get_localized_position(f)
15         if cur_pos ≠ None then
16             break;
17         end
18     end
19     P.append(cur_pos); /* cur_pos will be None if failed       */
20 end
```

Results

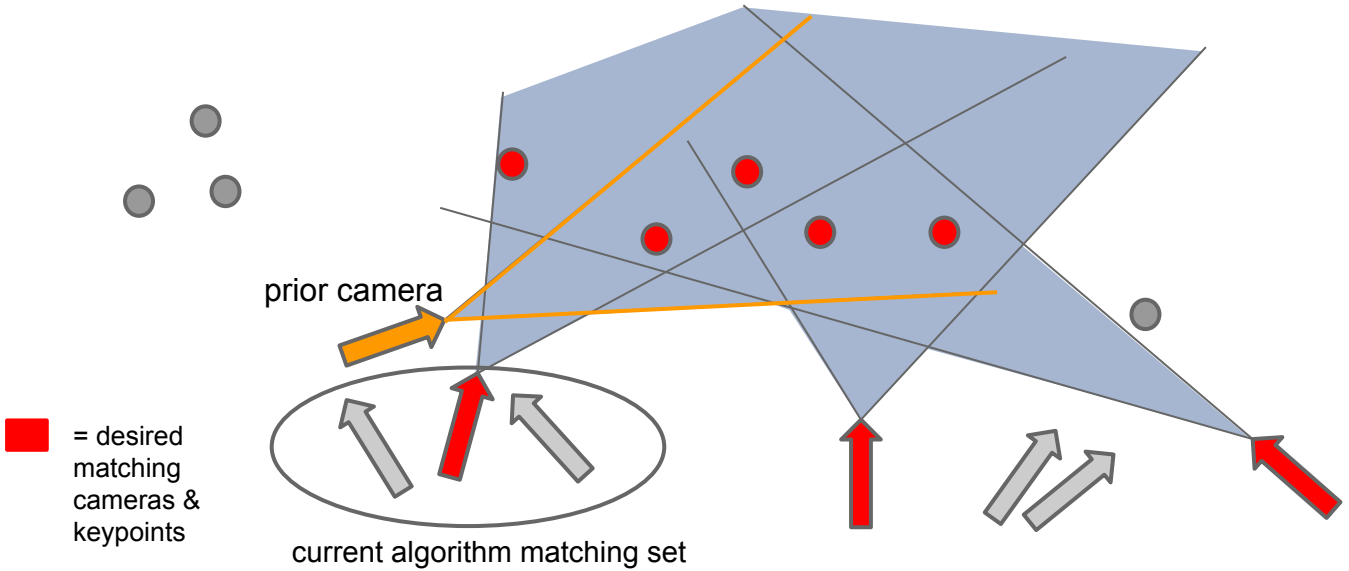
- Prior on camera location reduces matching set significantly (orders of magnitude)
- Speedup of full localization usually around 10x for model of ~1500 images (20 seconds -> 2 seconds on my computer), main speedup comes from not performing all-pairs matching. Slower in areas of model with “denser” cameras



Possible improvements

- **Goal 1 extension**: Piecewise planar approximation for larger SFM scenes composed of ground plane at multiple heights.

- **Goal 2 extension:** Localization could fail in SFM models with larger baselines / sparser, instead should match against the cameras that view keypoints nearby the keypoints of those recently viewed



References

[1] Furukawa, Yasutaka, and Jean Ponce. "Accurate, dense, and robust multiview stereopsis." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 32.8 (2010): 1362-1376.
[2] Hedau, Varsha, Derek Hoiem, and David Forsyth. "Recovering the spatial layout of cluttered rooms." *Computer vision, 2009 IEEE 12th international conference on*. IEEE, 2009.
[3] Wu, Changchang. "Towards linear-time incremental structure from motion." *3D Vision-3DV 2013, 2013 International Conference on*. IEEE, 2013.

Appendix (video links)

Ground planes:

- https://www.dropbox.com/s/tnvtp3retuumkw2/nsh4rotate_vis2.mp4?dl=0
- https://www.dropbox.com/s/w1qzln37zypc54h/smith2rotate_vis.mp4?dl=0

All ground planes:

- https://www.dropbox.com/s/jdffy11b4iudvdq/planes_all.mp4?dl=0