Randomized View Planning and Occlusion Removal for Mosaicing Building Facades

Christopher Rasmussen Thommen Korah William Ulrich

Department of Computer & Information Sciences
University of Delaware
{cer,tkorah,ulrich}@cis.udel.edu

***Abstract*— We present two key parts of an ongoing robotic, vision-based architectural modeling project. The first component is a randomized approach to view planning for a single ground robot scanning a building perimeter to recover a series of texture map mosaics. This algorithm generates paths that simultaneously address coverage and quality (i.e., real-valued distance and foreshortening factors). The second part is a technique for "cleaning" the captured texture maps in the presence of occlusions caused by trees, signs, people, and other foreground objects. When such occlusions comprise a minority of views a background feature can be recovered via temporal median filtering, but when they are in the majority, appearance information from other visible portions of the facade provides a critical cue to correctly complete the mosaic. We describe a novel spatiotemporal *timeline-based* inpainting algorithm that identifies and corrects such areas.**

I. Introduction

As part of a vision-based architectural modeling project (see [1], [2] for related work by others), we seek to capture the visual appearance of building exteriors via "scanning" by a single camera-equipped ground robot. Given an *a priori* polyhedral model of a building's structure (with assumed vertical, planar walls), a major subgoal of the task is to obtain a visual texture map of each section of its facade. Two large issues must be addressed to accomplish this:

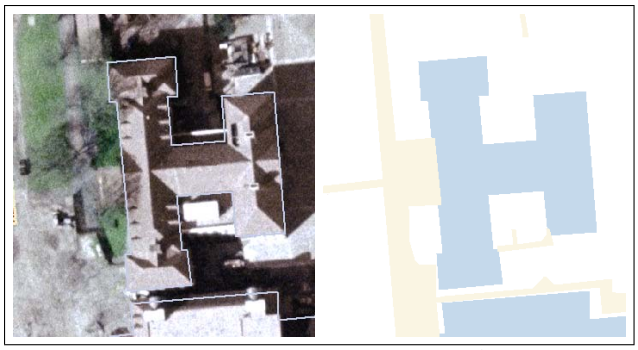
- View planning: How best to move the robot in order to rapidly achieve complete visual coverage
- Mosaicing: How to stitch together images taken from different positions, including filtering out pedestrians, trees, and other foreground objects which should not be included in the building appearance model

View Planning Our algorithm for facade texture map acquisition (see below) operates on a discrete set of overlapping views. Planning a robot path around the outside of a building that maximizes visual coverage (an aerial photo of an example building and its polygonal outline is shown in Fig. 1) is related to the "art gallery" problem from computational geometry [3]. Specifically, the task is to find a set of "guard" positions $\mathcal{G}$ in a polygon P that collectively "see" the entire polygon. The traditional criterion for *visibility* between two points $\mathbf{p}$ and $\mathbf{q}$ is *line of sight*: the line segment joining them does not intersect P. Paths along which the entire polygon is seen at least once are called *watchman routes*.

Robotics researchers interested in view planning with real sensors such as laser range-finders have recently extended the notion of visibility in an art gallery framework to include a *range constraint* and an *incidence constraint* [4], [5]. The range constraint models a sensor's inability to work when too far from or too close to an object. Points $\mathbf{p}$ on P whose distance from the sensor position $\mathbf{q}$ falls inside a specific range $d_{\min} \leq d(\mathbf{p}, \mathbf{q}) \leq d_{\max}$ are "range visible". Similarly, the incidence constraint enforces an angular range to model a limited sensor field of view (or exclusion of poor quality range returns at near-grazing angles). Letting $\mathbf{v} = \mathbf{q} - \mathbf{p}$ and $\mathbf{n}$ be the surface normal at $\mathbf{p}$, points for which the angle $\angle(\mathbf{n}, \mathbf{v}) \leq \tau_{\angle}$ are considered "incidence visible." Only polygon points which are visible in all three of the above senses are considered visible from a position $\mathbf{q}$.

Binary line-of-sight and field of view constraints obviously apply to camera view planning, but even within a single image different pixels "see" more or less of the building and with different levels of goodness. Horizontal and vertical foreshortening and finite camera resolution influence the overall goodness of a hypothetical view, and we believe that a real-valued visibility function—i.e., how well, not just whether, a particular point is seen—can better represent this. One contribution of this paper is a formulation for such a goodness function, which we demonstrate in an existing point sampling framework. Moreover, overlap between adjacent frames is important for mosaicing. To this end, we introduce an online method derived from particle filtering for finding and linking together guard points that allows dynamics and may be suitable for situations in which *a priori* building maps are not available (i.e., exploration). Finally, we offer some heuristics for improving the roadmap approach to extracting a path from the set of guard points described in [4] that are particular to our task.

Mosaicing Given a sequence of overlapping images of a single large plane of a building wall taken along the planned path, we aim to reconstruct an accurate map of that section of the facade. Creating a planar mosaic via homography estimation has been thoroughly studied [6], [7]. The complicating factor that motivates the second part of this paper is the possible presence of other, unknown objects in the scene between the camera and building plane—e.g., trees, people, signs, poles,



(a) (b)

Fig. 1. (a) Aerial view of example building to be mosaiced and its surroundings; (b) Outline of building and neighbor, plus additional map features (freespace here is of course the *outside* of the building polygons)

and other clutter of urban environments. Without explicitly recognizing them, these objects may be erroneously included in the building appearance model. Robust estimation of the dominant motion of the building converts the problem of "occluder removal" to a *foreground subtraction* or *layer extraction* problem [8], [9], [10], [11]. Many of these approaches either assume that the moving objects are relatively small compared to the background, facilitating temporal median filtering [12], [8], or that the objects to be removed are manually identified once [13], [9] in order to segment them later.

In recent work [14], we detailed a comprehensive approach to automatically identifying foreground regions and removing them via image/video inpainting [15], [16]. Pure inpainting is strictly necessary only where the background is never seen for the entire sequence, but our chief innovation is using regions visible in at least one view to constrain what should be painted there. By combining spatial information from pixels in a partially-completed mosaic with the temporal cues provided by images in the *timeline*, or sequence of images captured, sequences that present significant difficulties for temporal median filtering can be well-handled.

In the sections that follow, we will detail our randomized technique for camera view planning, estimation of foreground likelihoods over the sequence of images captured along the path, and integration of this information into a spatiotemporal inpainting algorithm built upon the non-parametric method described by Criminisi *et al.* in [16].

II. View Planning

Suppose we wish to mosaic or *cover* the exterior of a building B with camera views. B has a known perimeter and facade height h_B, and is one of a set of neighboring buildings $\mathcal{B} = \{B_1, \ldots, B_n\}$. Let the robot be equipped with a panning camera that has a vertical field of view of ϕ, a horizontal field of view of θ, and an image resolution of $w \times h$. We assume that the camera is mounted near ground level with its optical axis fixed parallel to the ground plane (i.e., it does not tilt). Image pixels discretize viewing directions, so for a cylindrical projection B's visibility is sampled by w equally-spaced viewing rays.

A. Goodness Function

We replace the binary visibility criterion above with a real-valued "goodness" function. Consider a candidate viewing position $\mathbf{q}$ in the plane. A particular viewing ray is defined by $\mathbf{q}$ and a unit direction vector $\mathbf{r}$. The goodness of the ray $\Gamma(\mathbf{q}, \mathbf{r})$ is defined as the product of the $\{0, 1\}$-valued line-of-sight, range, and incidence constraints ($\tau_{\angle} = \theta/2$) defined above and two real-valued terms (only calculated if the first three are all non-zero). These are:

- **Foreshortening** If the normal at the wall point $\mathbf{p}$ on B which the ray strikes is $\mathbf{n}$, the dot product $-\mathbf{r} \cdot \mathbf{n}$ is 1 when the ray hits the building orthogonally and 0 at the extreme grazing angle of 90 degrees. This measures the effective resolution of the pixel.
- **Vertical Framing** An additional measure of pixel utilization is how well the building fills the image vertically. We penalize for being too far away, resulting in sky visible above the building, as well as being too close, cutting off the top of the building. If $\mathbf{p}$ is on the ground, another point $\mathbf{p}'$ that is $h_{\mathbf{p}'} = d(\mathbf{p}, \mathbf{q}) \tan \phi$ meters above the ground would be in the top row of the image. Thus, $\min(h_{\mathbf{p}'}, h_B) / \max(h_{\mathbf{p}'}, h_B)$ measures the vertical fraction of the image that is either sky or cut-off building.

The goodness of a viewing position $\Gamma(\mathbf{q})$ is evaluated by casting one ray per sensor pixel and taking their average goodness. In this work we ignore vertical foreshortening by only casting rays in the plane—i.e., horizontally. A sample synthetic omnidirectional image (only 270 degrees are shown) is given in Fig. 2(a). It represents the information available to the planner regarding visibility, foreshortening, and vertical framing via the intensity of each image pixel. Fig. 2(b) shows a building surrounded by sampled viewing positions (more on this in the next subsection) and a particular position at which Γ is being calculated. The intensities of the rays are proportional to their individual goodnesses. A high-resolution depiction of the components of the goodness objective function is shown in Fig. 2(c-e).

B. Next Best View

Using the binary visibility criteria of distance and angle defined above, the first algorithm in [5] based on the Greedy algorithm for finding near-optimal set covers could be directly used to obtain a set of guards $G_1, G_2, \ldots$ that covers the building polygon B with views. Briefly, one would randomly and uniformly sample viewing positions outside any building polygon and within $d_{\max}$ of B, compute what sections of B's perimeter are visible from each sample, and pick the guard position that sees the longest overall section of B (summed over visible fragments). These "already seen" sections are marked as invisible to subsequent viewing position candidates, and the process is repeated until all of B (up to some threshold) has been seen once.

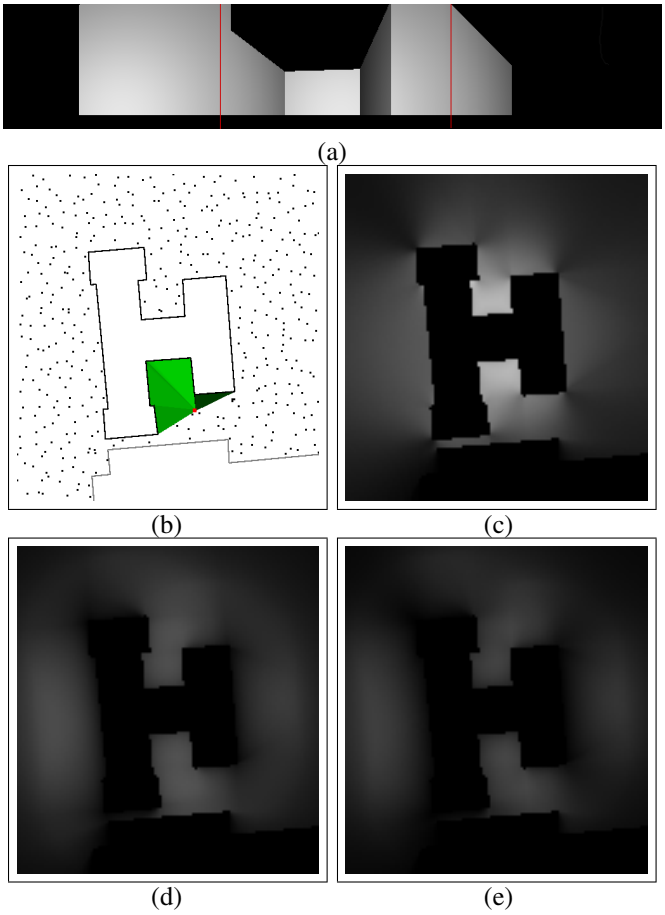


Fig. 2. (a) Sample synthetic omnidirectional image (partial) of building with diffuse shading showing foreshortening (red lines are 90 degree intervals); (b) Building polygon, uniform samples, and viewing position with cast rays (ray saturation proportional to goodness); (c) Goodness function with foreshortening term only discretized at 1 m resolution; (d) Goodness function with vertical framing term only; (e) Combined (via product) goodness function

The real-valued goodness function above necessitates modifications to this approach since each section of wall is not just "viewed" or "not viewed", but rather "viewed with a certain goodness." Therefore, we set a threshold on the *total goodness* with which every segment of wall must be viewed. The bookkeeping for such a requirement makes the exact ray-sweeping methods used in GREEDY inapplicable. Therefore, we discretize the perimeter of the building into initially empty buckets and every new guard chosen deposits its ray goodnesses into the buckets until their thresholds are exceeded, after which subsequent rays hitting such areas have goodness 0. This mechanism ensures as before that new guard positions are chosen rather than the same one repeatedly, and updating this data structure is extremely simple and fast. We call this variant GOODNESSGREEDY .

C. Path Building

The set of guards obtained with the GOODNESSGREEDY method above is unordered and not immediately useable as a path for robot motion. We have developed two approaches to making this step.

1) Uniform Sampling: Previous work [4] demonstrated how post-processing could generate an ordering of the guard points using an approximate approach to the traveling salesman problem. A graph R called a "roadmap" is created which contains all guard points and building vertices, with edges joining mutually visible nodes. The shortest path between every pair of guards in R is then computed to yield a fully connected graph R^*. A traversal of R^*'s minimum spanning tree R^*_{MST} yields an "inspection tour" with length less than or equal to twice that of the shortest possible tour.

We have implemented a variant of this approach using several heuristics in the traversal of R^*_{MST} to shorten the extracted path. Rather than [4]'s method of simply performing a pre-order traversal, we initially choose a path start point that maximizes tree height, and then at each node with multiple children we visit the children in order from shortest to tallest subtree height. These heuristics tend to work well for circumnavigating a building with concavities because exploring such indentations are just small detours off of the mainline circuit around the building. Typically, we smooth the final path to eliminate sharp corners for better suitability to robot motor control using an interpolating spline such as Catmull-Rom [17].

2) Particle Filter: An alternative, online approach to building a robot inspection path is derived from the idea of reactive path planning via a "tracking quality" function in [18]. Here the idea is to define a "map quality" function on robot positions that is just the real-valued goodness function above.

Given an initial robot position in the area near a building, we sample positions in its neighborhood with a Gaussian in a manner similar to a particle filter [19]. The local sample with the best positional goodness is chosen as the next guard, and the area around it is sampled. Through segments of the building bucket data structure filling up, the path planner is "forced" to move in order to find novel views. This effectively builds a path online for the robot through randomized gradient ascent that constitutes coverage behavior without explicitly building and searching a roadmap graph. Because the sampling is local, of course, the planner can become temporarily stuck in local minima until it random-walks to a new view. In practice we have found that this approach works quite well even on buildings or sets of buildings with many concavities.

III. TEXTURE MAP ACQUISITION

Given a set of overlapping images captured from the viewing positions planned in the previous section, we wish to build a mosaic of each planar wall section. Assuming the captured images have been grouped into sets corresponding to planar wall sections, we work on each group separately.

A. Image registration

We begin by computing the dominant planar motion (assumed to belong to one side of the building) between successive pairs of images $\mathbf{I}_t, \mathbf{I}_{t+1}$ in a sequence of N frames. These initial frame-to-frame homographies $\mathbf{H}^*_{t,t+1}$ are computed by

matching KLT features [20] in both frames followed by RANSAC for outlier rejection [21] . Taking frame number $ref = \lceil \frac{N}{2} \rceil$ of the sequence as the *mosaic reference frame*, the homographies are then concatenated together to align each frame with the mosaic—i.e., $\mathbf{H}^*_{ref,ref}$ is the identity; for $t < ref$, $\mathbf{H}^*_{t,ref} = \mathbf{H}^*_{ref-1,ref} \cdots \mathbf{H}^*_{t+1,t+2}\mathbf{H}^*_{t,t+1}$; and similarly for $t > ref$. Warping each frame $\mathbf{I}_t$ by $\mathbf{H}^*_{t,ref}$ with bilinear interpolation results in a mosaic-aligned frame $\mathbf{W}^*_t$.

Computing frame-to-mosaic homographies this way worsens misalignment errors for frames distant from the reference. With additional constraints on frame alignments (e.g., that the first and last or other temporally distant image pairs overlap), global consistency methods [22] or other forms of *bundle adjustment* may mitigate such errors. With a 1-D scanning motion around the building perimeter we cannot take advantage of these methods. Thus, we minimize alignment errors by refining the initial feature-based homographies with a robust direct method that iteratively minimizes the sum of squared differences (SSD) between frames [23], [24]. This procedure operates sequentially on adjacent pairs of warped images $\mathbf{W}^*_i, \mathbf{W}^*_j$ starting from $\mathbf{W}^*_{ref}$ and working outward. After concatenating these refined pairwise homographies, we obtain a final set of refined frame-to-mosaic homographies $\mathbf{H}_{t,ref}$ and stabilized images $\mathbf{W}_t$.

B. Identifying Problem Pixels

Each location $\mathbf{p} = (x, y)$ in the mosaic reference frame has a set of pixels from the warped images $\{\mathbf{W}_t(\mathbf{p})\}$ associated with it which we call its *timeline* $\mathcal{T}(\mathbf{p})$. The size of each timeline $|\mathcal{T}(\mathbf{p})|$ may vary from 0 to N depending whether the pixel at $\mathbf{p}$ was imaged or not in each frame. Intuitively, since all pixels on the building facade exhibit the dominant motion, they should appear stationary in the mosaic whereas foreground objects such as trees and signs move due to parallax. Given that each $\mathcal{T}(\mathbf{p})$ contains an unknown mixture of background and foreground object pixels, our goal is to correctly pick or estimate each background pixel $\mathbf{M}(\mathbf{p})$ where $|\mathcal{T}(\mathbf{p})| > 0$, forming a building mosaic $\mathbf{M}$. In this paper we assume that the lateral and vertical limits of the building associated with corners, the roofline, the ground, etc. are given, and we do not rectify the mosaic to compensate for an oblique viewing angle.

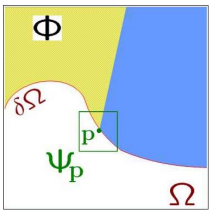


Fig. 3. Source region Φ, target region Ω, target boundary $d\Omega$, target patch $\Psi_{\mathbf{p}}$ (from Criminisi *et al.* [16])

A robust estimator for $\mathbf{M}(\mathbf{p})$ under the assumption that foreground pixels are in the minority (i.e., outliers) in $\mathcal{T}(\mathbf{p})$ is the *temporal median* $\mathbf{M}(\mathbf{p}) = median(\mathcal{T}(\mathbf{p}))$. This is computed separately for each color channel: $\mathbf{M}_{red}(\mathbf{p}) = median(\mathcal{T}_{red}(\mathbf{p}))$, and so on, giving rise to the median mosaic $\mathbf{M}_{med}$. This estimator fails, however, when foreground pixels are in the majority in a particular timeline. We observe that except for large homogeneous foreground regions or *camouflaged* foreground objects with almost the same color as the background, the likelihood that $\mathcal{T}(\mathbf{p})$ has a majority of foreground pixels is proportional to the variability or "spread" of its color distribution. To robustly measure this variability, we use the median absolute deviation (MAD) [25], defined as $MAD(\mathcal{T}(\mathbf{p})) = median(|\mathbf{W}_t(\mathbf{p}) - median(\mathcal{T}(\mathbf{p}))|)$ over all t in the timeline. A scalar MAD value is obtained at each pixel by computing it separately for each color channel and summing. A high MAD value at $\mathbf{p}$ indicates a higher likelihood that $\mathbf{M}_{med}(\mathbf{p})$ is unreliable, so unreliable median mosaic pixels are filtered out by thresholding their MADs—these are so-called *MAD outlier* pixels. Finally, the raw MAD outlier mask is spatially smoothed with a morphological majority operation.

C. Timeline Inpainting

In this section we present an algorithm for filling the MAD outliers in $\mathbf{M}_{med}$ that is built upon the work in Criminisi, Pérez, and Toyama [16], a patch-based copying method combining ideas from non-parametric texture synthesis and diffusion-based inpainting. We will refer to their method as *CPT inpainting* (reviewed in detail in [14]) vs. ours of *timeline inpainting*. As shown in Fig. 3, in CPT inpainting an empty target region Ω's pixels are filled from its border $d\Omega$ inward by copying square image patches from a source region Φ to target patches $\Psi_{\mathbf{p}}$ centered on $\mathbf{p} = (x, y) \in d\Omega$.

Let the MAD outlier pixels be the target region Ω and the rest of the median mosaic $\mathbf{M}_{med}$ be the source region Φ. Our problem differs from pure spatial inpainting in that the timeline $\mathcal{T}$ for each $\mathbf{p} \in \Omega$, provided it contains at least one background pixel, should constrain the filling process. Thus, our major goals are to determine which, if any, pixels in $\mathcal{T}(\mathbf{p})$ are from the building background, and to integrate this information into the inpainting process. Letting $\mathcal{T}(\Psi_{\mathbf{p}}) = \{\Psi^1_{\mathbf{p}}, \ldots, \Psi^{|\mathcal{T}(\mathbf{p})|}_{\mathbf{p}}\}$ be the timeline of patches centered on $\mathbf{p}$, we create a *timeline mosaic* $\mathbf{M}_{time}$ by modifying CPT inpainting in three major ways:

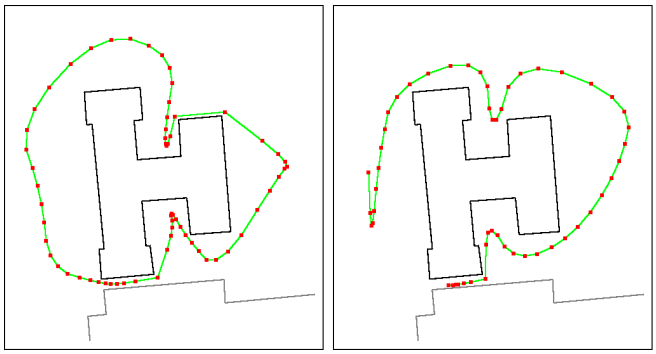
1) In the first of two stages, each patch-wise pixel copy to Ω comes *from one timeline patch* $\Psi^*_{\hat{\mathbf{p}}} \in \mathcal{T}(\Psi_{\hat{\mathbf{p}}})$ maximally likely to have come from the building
2) During stage one, the updated confidences $C(\mathbf{p})$ of newly-filled pixels are set to the motion-based *background likelihoods* $p^*_{back}(\mathbf{p})$ of the pixels in $\Psi^*_{\hat{\mathbf{p}}}$
3) If the mean background likelihood $\bar{p}_{back}(\Psi^t_{\hat{\mathbf{p}}})$ for every patch in $\mathcal{T}(\Psi_{\hat{\mathbf{p}}})$ is below a threshold τ_{back}, $\Psi_{\hat{\mathbf{p}}}$ is *not filled* at that time. Stage two begins when all remaining areas of Ω meet this definition, and consists simply of CPT inpainting

Each of these three modifications is explained in more detail in [14].

IV. Results

A. View planning

We have performed extensive experiments on large, concave building polygons, both individually and in sets, and observed excellent performance for both the particle filter variant described above and the modified version of the uniform



(a) (b)

Fig. 4. Sample planned coverage paths (subdivided). (a) Using global, GOODNESSGREEDY method for guard point selection followed by a distinct guard point linking phase; (b) Using local, particle filter-like algorithm to incrementally add guard points from a random start point

Fig. 5. Raw frames from building sequence

sampling algorithm from [5]. In the limited space available here, we show some generated paths for a building using each method in Fig. 4. 360 rays (1 per degree) were cast per sample position, the goodness bucket width along the building outline was 2 m (the building perimeter is about 325 m), and the vertical FOV ϕ was 30 degrees. For uniform sampling, one sample was generated per 5×5 m square ($d_{\min} = 1$ m, $d_{\max} = 50$ m), while for the particle filter-like approach 300 particles were used, generated from a normal distribution with variance 150.

B. Mosaicing

We describe the operation of our algorithm on a preliminary image sequence. 801 24-bit color frames, resampled to 360×240 pixels each, were captured at 30 fps from a manually-controlled camera moving parallel to a building facade. Several objects at different depths occlude parts of the building including trees, bushes, and a large sign. Our algorithm was run on a subset of 17 frames from the sequence taken at intervals of every 50 frames; the first and last of these are shown in Fig. 5.

The median mosaic $\mathbf{M}_{med}$ shown in Fig. 6(a) is mostly quite good, recovering almost all of the facade cleanly. The near tree (e.g., the last frame in Fig. 5) is almost entirely removed (some artifacts near the mosaic edges are due to an insufficient number of overlapping images there). This is because its large parallax motion causes occlusions to be brief and thus tree pixels are in the minority in the timeline vs. building pixels. A significant problem area, however, is created by the more distant tree, which exhibits relatively little parallax motion. This object occludes many building pixels in a majority of frames, confounding the median filter.

Areas where $\mathbf{M}_{med}$ is poor correlate well with the MAD outliers. The result of a conservative threshold which tags about 20% of pixels as outliers is shown in Fig. 6(b). CPT inpainting to fill Ω_0 is insufficient, as too much structure is hidden. The results after stage one of timeline inpainting are shown in Fig. 6(c). For this stage, a shifting, circular search region $\Phi(\mathbf{p})$ (radius $= 150$ pixels) around each 11×11 patch's center $\mathbf{p}$ was used. The results after CPT inpainting in stage two are shown in Fig. 6(d). Spatial inpainting tends to introduce some misalignment artifacts, but ongoing work indicates that these will be lessened by correcting for radial distortion and rectifying the image before applying our algorithm.

V. CONCLUSION

We have presented two parts of an autonomous architectural modeling system involving view planning and mosaic construction. The first is a randomized view planner that efficiently builds incremental paths that maximize coverage and quality. The second part is a novel approach to detecting and removing occlusions of building facades in image sequences using a combination of temporal and spatial inpainting. We are currently investigating techniques for straight line analysis and vanishing point detection [2] to automatically rectify the texture map and segment planar facade regions from the ground and each other at building corners.

Fig. 7. Architectural modeling robot

It is important to note that the mosaicing sequence in the Results section was manually captured along a path segment that is stereotypical but not the result of our planning algorithm. Our current work focuses on integrating and testing the software components described here on a recently-built robot, shown in Fig. 7. Following a design concept pioneered by researchers at the University of Pennsylvania [26], the University of Southern California [27], and elsewhere, our robot platform is based on a stock 1/10 scale radio-controlled (RC) truck chassis.

The prototype robot is equipped with a pan-tilt-zoom (PTZ) color CCD camera with an analog video capture interface, a global positioning system (GPS) receiver and antenna, and a precision 3-axis orientation sensor and digital compass. The PTZ camera allows the camera view to be set independently of the robot heading, with the orientation sensor and the GPS together giving a relatively accurate estimate of the 3-D position and 3-D orientation of the robot's camera in world coordinates. Motor and CPU battery life currently allow autonomous operation for 45-60 minutes at a time.

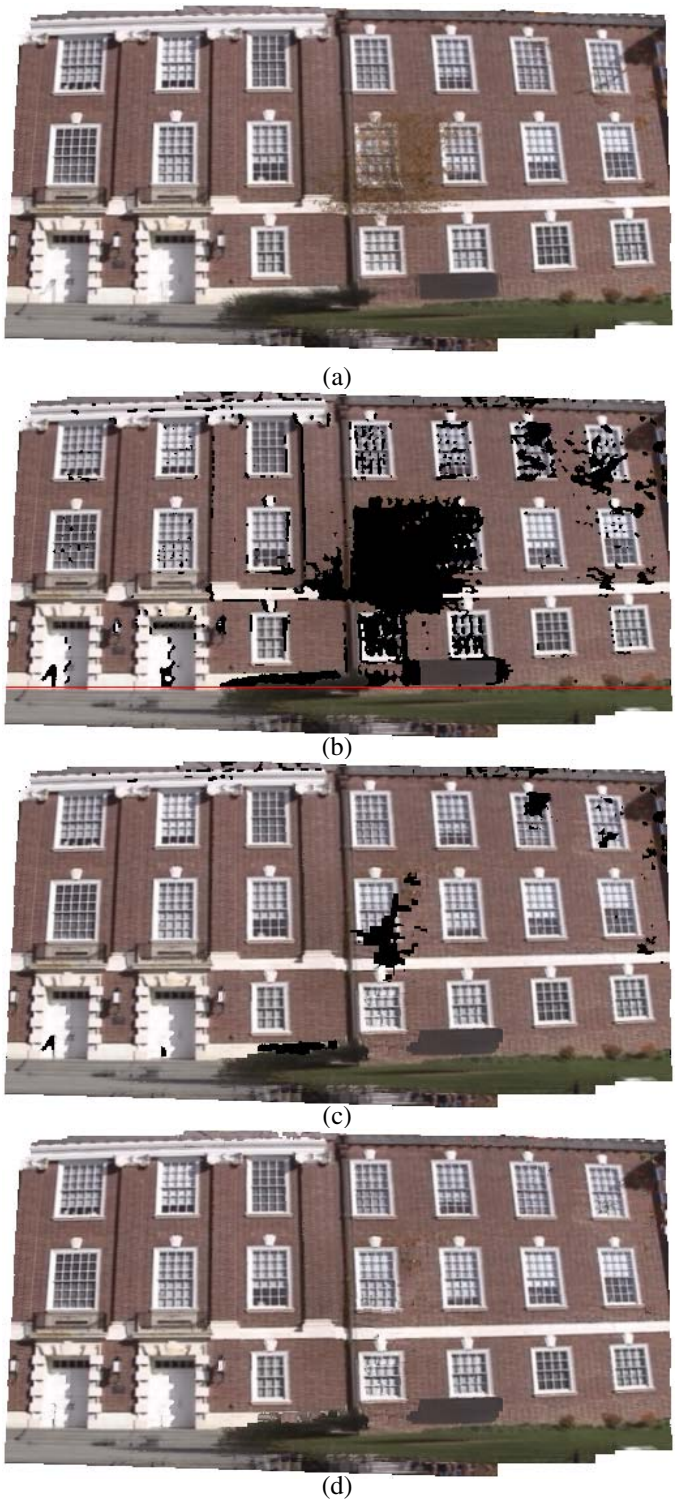


Fig. 6. (a) Median mosaic $\mathbf{M}_{med}$; (b) MAD outliers in $\mathbf{M}_{med}$; (c) $\mathbf{M}_{time}$ after stage one; (d) $\mathbf{M}_{time}$ after stage two

ACKNOWLEDGMENTS

This work was supported by a grant from the University of Delaware Research Foundation.

REFERENCES

[1] S. Teller, M. Antone, Z. Bodnar, M. Bosse, S. Coorg, M. Jethwa, and N. Master, "Calibrated, registered images of an extended urban area," *Int. J. Computer Vision*, 2003.

[2] F. van den Heuvel, "Automation in architectural photogrammetry; line-photogrammetry for the reconstruction from single and multiple images," Ph.D. dissertation, Delft University of Technology, Delft, The Netherlands, 2003.

[3] J. O'Rourke, *Art Gallery Theorems and Algorithms*. Oxford University Press, 1987.

[4] T. Danner and L. Kavraki, "Randomized planning for short inspection paths," in *Proc. Int. Conf. Robotics and Automation*, 2000.

[5] H. Gonzalez-Banos and J. Latombe, "A randomized art-gallery algorithm for sensor placement," in *Proc. 17th ACM Symp. on Computational Geometry*, 2001.

[6] M. Hansen, P. Anandan, K. Dana, G. van der Wal, and P. Burt, "Real-time scene stabilization and mosaic construction," in *DARPA Image Understanding Workshop*, 1994.

[7] R. Szeliski, "Video mosaics for virtual environments," *IEEE Computer Graphics and Applications*, vol. 16, no. 2, pp. 22–30, 1996.

[8] F. Odone, A. Fusiello, and E. Trucco, "Layered representation of a video shot with mosaicing," *Pattern Analysis and Applications*, vol. 5, pp. 296–305, 2002.

[9] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut - interactive foreground extraction using iterated graph cuts," in *SIGGRAPH*, 2004.

[10] J. Wang and E. Adelson, "Representing moving images with layers," *IEEE Trans. Image Processing*, vol. 3, no. 5, 1994.

[11] A. Fitzgibbon, Y. Wexler, and A. Zisserman, "Image-based rendering using image-based priors," in *Proc. Int. Conf. Computer Vision*, 2003.

[12] D. Farin, P. de With, and W. Effelsberg, "Robust background estimation for complex video sequences," in *Proc. IEEE Int. Conf. on Image Processing*, 1997.

[13] A. Kokaram, B. Collis, and S. Robinson, "A bayesian framework for recursive object removal in movie post-production," in *Proc. IEEE Int. Conf. on Image Processing*, 2003.

[14] C. Rasmussen and T. Korah, "Spatiotemporal inpainting for recovering texture maps of partially occluded building facades," in *Submitted to IEEE Int. Conf. on Image Processing*, 2005.

[15] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *SIGGRAPH*, 2000, pp. 417–424.

[16] A. Criminisi, P. Pérez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Processing*, vol. 13, no. 9, 2004.

[17] J. Hoschek and D. Lasser, *Fundamentals of Computer-Aided Geometric Design*. A.K. Peters, 1993.

[18] J. Spletzer and C. Taylor, "A framework for sensor planning and control with applications to vision guided multi-robot systems," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2001.

[19] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *Proc. European Conf. Computer Vision*, 1996, pp. 343–356.

[20] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1994.

[21] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[22] H. Sawhney, S. Hsu, and R. Kumar, "Robust video mosaicing through topology inference and local to global alignment," in *Proc. European Conf. Computer Vision*, 1998.

[23] M. Irani, B. Rousso, and S. Peleg, "Computing occluding and transparent motions," *Int. J. Computer Vision*, 1994.

[24] M. Black and P. Anandan, "The robust estimation of multiple motions: parametric and piecewise-smooth flow fields," *Computer Vision and Image Understanding*, 1996.

[25] T. Tommasini, A. Fusiello, E. Trucco, and V. Roberto, "Making good features to track better," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1998, pp. 178–183.

[26] R. Alur, A. Das, J. Esposito, R. Fierro, Y. Hur, G. Grudic, V. Kumar, I. Lee, J. P. Ostrowski, G. Pappas, J. Southall, J. Spletzer, and C. Taylor, "A framework and architecture for multirobot coordination," in *Proc. Int. Symposium on Experimental Robotics*, 2001.

[27] E. Pichon and L. Itti, "Real-time high-performance attention focusing for outdoors mobile beobots," in *Proc. AAAI Spring Symposium, Stanford, CA (AAAI-TR-SS-02-04)*, Mar 2002, p. 63.