Integrating Automated Range Registration with Multiview Geometry for the Photorealistic Modeling of Large-Scale Scenes

Ioannis Stamos[†], Lingyun Liu[†], Chao Chen[†], George Wolberg[‡], Gene Yu[‡], Siavash Zokai[††]

[†]*Dept. of Computer Science, Hunter College / CUNY, New York, NY, USA; and*
[‡]*Dept. of Computer Science, City College of New York / CUNY, New York, NY, USA; and*
[††]*Brainstorm Technology LLC, New York, NY, USA*
E-mail: istamos@hunter.cuny.edu, lingyun77@gmail.com, cchen@gc.cuny.edu, wolberg@cs.ccny.cuny.edu

The photorealistic modeling of large-scale scenes, such as urban structures, requires a fusion of range sensing technology and traditional digital photography. This paper presents a system that integrates automated 3D-to-3D and 2D-to-3D registration techniques, with multiview geometry for the photorealistic modeling of urban scenes. The 3D range scans are registered using our automated 3D-to-3D registration method that matches 3D features (linear or circular) in the range images. A subset of the 2D photographs are then aligned with the 3D model using our automated 2D-to-3D registration algorithm that matches linear features between the range scans and the photographs. Finally, the 2D photographs are used to generate a second 3D model of the scene that consists of a sparse 3D point cloud, produced by applying a multiview geometry (structure-from-motion) algorithm directly on a sequence of 2D photographs. The last part of this paper introduces a novel algorithm for automatically recovering the rotation, scale, and translation that best aligns the dense and sparse models. This alignment is necessary to enable the photographs to be optimally texture mapped onto the dense model. The contribution of this work is that it merges the benefits of multiview geometry with automated registration of 3D range scans to produce photorealistic models with minimal human interaction. We present results from experiments in large-scale urban scenes.

1. INTRODUCTION

The photorealistic modeling of large-scale scenes, such as urban structures, requires a combination of range sensing technology with traditional digital photography. A systematic way for registering 3D range scans and 2D images is thus essential. This paper presents a system that integrates multiview geometry and automated 3D registration techniques for texture mapping 2D images onto 3D range data. The novelty of our approach is that it exploits all possible relationships between 3D range scans and 2D images by performing 3D-to-3D range registration, 2D-to-3D image-to-range registration, and structure from motion. Several papers, including this one, provide frameworks for automated texture mapping onto 3D range scans [23, 29, 42, 46, 53]. These methods are based on extracting features (e.g., points, lines, edges, rectangles or rectangular parallelepipeds) and matching them between the 2D images and the 3D range scans. Our approach provides a solution of increased robustness, efficiency and generality with respect to previous methods. Our contribution is discussed in Sec. 2.

Despite the advantages of feature-based texture mapping solutions, most systems that attempt to recreate photorealistic models do so by requiring the manual selection of features among the 2D images and the 3D range scans, or by rigidly attaching a camera onto the range scanner and thereby fixing the relative position and orientation of the two sensors with respect to each other [16, 36, 41, 48, 52]. The fixed-relative position approach provides a solution that has the following major limitations:

1. The acquisition of the images and range scans occur at the same point in time and from the same location in space. This leads to a lack of 2D sensing flexibility since the limitations of 3D range sensor positioning, such as standoff distance and maximum distance, will cause constraints on the placement of the camera. Also, the images may need to be captured at different times, particularly if there were poor lighting conditions at the time that the range scans were acquired.
2. The static arrangement of 3D and 2D sensors prevents the camera from being dynamically adjusted to the requirements of each particular scene. As a result, the focal length and relative position must remain fixed.
3. The fixed-relative position approach cannot handle the case of mapping historical photographs on the models or of mapping images captured at different instances in time. These are capabilities that our method achieves.

In summary, fixing the relative position between the 3D range and 2D image sensors sacrifices the flexibility of 2D image capture. Alternatively, methods that require manual interaction for the selection of matching features among the 3D scans and the 2D images are error-prone, slow, and not scalable to large datasets. Laser range scanning is a laborious, tedious, time consuming, and costly operation that precludes easy and cost-effective recapturing of data. Therefore, it is best to separate the geometry capture mode from the image acquisition mode so that the latter can be done quickly with fairly constant lighting conditions. These limitations motivate the work described in this paper, making it essential for producing photorealistic models of large-scale urban scenes.

The texture mapping solution described in this paper merges the benefits of multiview geometry with automated 3D-to-3D range registration and 2D-to-3D image-to-range registration to produce photorealistic models with minimal human interaction. The 3D range scans and the 2D photographs are respectively used to generate a pair of 3D models of the scene. The first model consists of a dense 3D point cloud, produced by using a 3D-to-3D registration method that matches 3D lines in the range images to bring them into a common reference frame. The second model consists of a sparse 3D point cloud, produced by applying a multiview geometry (structure-from-motion) algorithm directly on a sequence of 2D photographs to simultaneously recover the camera motion and the 3D positions of image features. This paper introduces a novel algorithm for automatically recovering the similarity transformation (rotation/scale/translation) that best aligns the sparse and dense models. This alignment is necessary to enable the photographs to be optimally texture mapped onto the dense model. No a priori knowledge about the camera poses relative to the 3D sensor's coordinate system is needed, other than the fact that one image frame should overlap the 3D structure (see Sec. 4). Given one sparse point cloud derived from the photographs and one dense point cloud produced by the range scanner, a similarity transformation between the two point clouds is computed in an automatic and efficient way. The framework of our system is shown in Fig. 1. Each of the framework elements listed below, is a distinct system module in Fig. 1.

- A set of 3D range scans of the scene is acquired and co-registered to produce a dense 3D point cloud in a common reference frame (Sec. 3).
- An independent sequence of 2D images is gathered, taken from various viewpoints that do not necessarily coincide with those of the range scanner. A sparse 3D point cloud is reconstructed from these images by using a structure-from-motion (SFM) algorithm (Sec. 5).
- A *subset* of the 2D images are automatically registered with the dense 3D point cloud acquired from the range scanner (Sec. 4).
- Finally, the *complete* set of 2D images is automatically aligned with the dense 3D point cloud (Sec. 6). This last step provides an integration of all the 2D and 3D data in the same frame of reference. It also provides the transformation that aligns the models gathered via range sensing and computed via structure from motion.

2. RELATED WORK

A robust method that extracts distinguishable features from range images is very important for our method. Previous range image segmentation techniques include edge detection [3, 49], region growing [5, 37], and polynomial surface fitting [5, 12]. Most of these methods provide edge maps and/or regions expressed as polynomial functions. This is useful for object modeling and reconstruction, but may not be suitable for feature matching. Our method detects precise edges and extracts geometric features with concise descriptors that make them appropriate for feature matching.

Iterative Closest Point (ICP) is one of the most popular range registration algorithms [6, 39]. ICP provides very accurate results but requires a good initial guess of the registration transformation. We, on the other hand, utilize ICP as a post-processing step after our automated method brings scans into alignment. A method that does not require knowledge of an initial registration transformation is presented in [21, 25] (spin images). The spin images approach does not rely on features of specific geometric type, but is sensitive to varying scan resolutions. Furthermore, the extracted point signatures have local support, the extent of which is specified by the user. There are many approaches for the solution of the pose estimation problem from both point correspondences [33, 38] and line correspondences [11, 20], when a set of matched 3D and 2D points or lines are known, respectively.

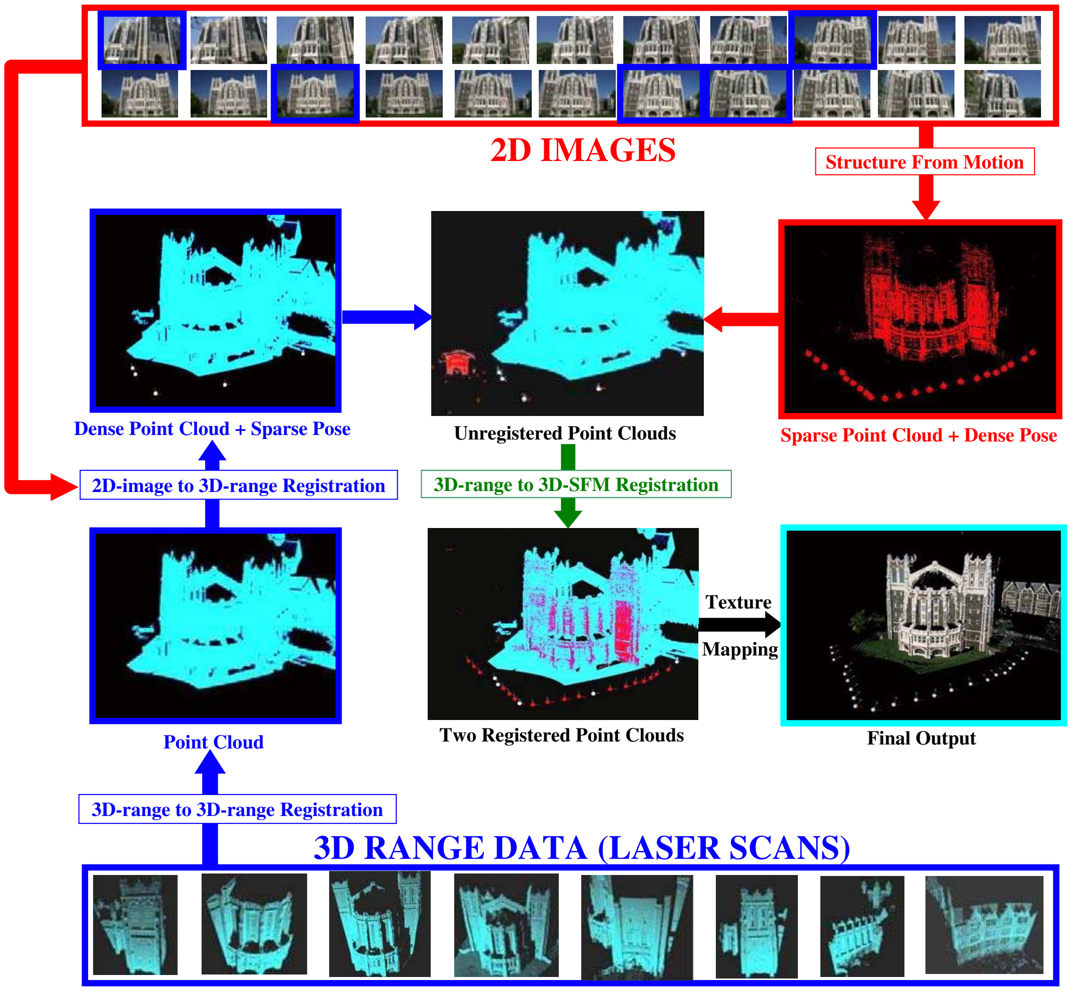

2D IMAGES
Structure From Motion
Dense Point Cloud + Sparse Pose
Unregistered Point Clouds
Sparse Point Cloud + Dense Pose
2D-image to 3D-range Registration
3D-range to 3D-SFM Registration
Texture
Mapping
Point Cloud
Two Registered Point Clouds
Final Output
3D-range to 3D-range Registration
3D RANGE DATA (LASER SCANS)

FIG. 1. System framework. Several registered range scans of Shepard Hall (CCNY) constitute a dense 3D point cloud model M_{range} shown in the leftmost column. The five white dots correspond to the locations of five of the 26 color images (shown as thumbnails on top row) that are independently registered with the model M_{range} via a 2D-to-3D image-to-range registration algorithm. The rightmost image of the second row depicts the 3D model M_{sfm} produced by SFM. The points of M_{sfm} as well as **all** the recovered camera positions for the sequence of 2D images that produced M_{sfm} are shown as red dots in the figure. Since SFM does not recover scale, M_{range} and M_{sfm} are not registered when brought to the same coordinate system, as shown in the second row. The 3D range model M_{range} overlaid with the 3D model M_{sfm} is shown in the third row of the figure after a 3D-range to 3D-SFM registration module aligns them together. The recovered camera positions from SFM can now be used to project the26 color images onto M_{range}, which now properly sits in the M_{sfm} coordinate system, to produce the richly textured 3D model (Final Output) shown in the right column.

In the early work of [15], the probabilistic RANSAC method was introduced for automatically computing matching 3D and 2D points. RANSAC is a robust method and can handle a large number of outliers. It is possible that the method may fail if presented with degenerate data configurations. In such cases, RANSAC may fit the model to outliers instead of inliers. Solutions in automated matching of 3D with 2D features in the context of object recognition and localization include [7, 18, 22, 24, 26, 50]. Very few methods, though, attack the problem of automated alignment of images with dense point clouds derived from range scanners. This problem is of major importance for automated photorealistic reconstruction of large-scale scenes from range and image data. In [29, 42] two methods that exploit orthogonality constraints (rectangular features and vanishing points) in man-made scenes are presented. The methods can provide excellent results, but will fail in the absence of a sufficient number of linear features. Ikeuchi [23], on the other hand, presents an automated 2D-to-3D registration method that relies on the reflectance range image. However, the algorithm requires an initial estimate of the image-to-range alignment in order to converge. Finally, [46] presents a method that works under specific outdoor lighting situations. A system whose goals are very similar to ours is described in [53]. In that work, continuous video is aligned onto a 3D point cloud obtained from a 3D sensor. First, an SFM/stereo algorithm produces a 3D point cloud from the video sequence. This point cloud is then registered to the 3D point cloud acquired from the range scanner by applying the ICP algorithm [6]. One limitation of this approach has to do with the shortcomings of the ICP algorithm. In particular, the 3D point clouds must be manually brought close to each to yield a good initial estimate that is required for the ICP algorithm to work. The ICP may fail in scenes with few discontinuities, such as those replete with planar or cylindrical structures. Also, in order for the ICP algorithm to work, a very dense model from the video sequence must be generated. This means that the method of [53] is restricted to video sequences, which limits the resolution of the 2D imagery. Finally, that method does not automatically compute the difference in scale between the range model and the recovered SFM/stereo model.

Our contributions can be summarized as follows:

- We automatically register the 3D range scans by matching linear and circular features.
- Like [53], we compute a model from a collection of images via SFM. Our method for aligning the range and SFM models, described in Sec. 6, does not rely on ICP and thus does not suffer from its limitations.
- We are able to automatically compute the scale difference between the range and SFM models.
- We perform 2D-to-3D image-to-range registration [28, 29] for a few (at least one) images of our collection. This feature-based method provides excellent results in the presence of a sufficient number of linear features. Therefore, the images that contain enough linear features are registered using that method. The utilization of the SFM model allows us to align the remaining images with a method that involves robust point (and not line) correspondences.
- We generate an optimal texture mapping result by using contributions of all 2D images.

3. 3D-TO-3D RANGE REGISTRATION

The first step is to acquire a set of range scans $S_i (i = 1, \ldots, K)$ that adequately covers the 3D scene. The laser range scanner used in our work is a Leica HDS 2500 [27], an active sensor that sweeps an eye-safe laser beam across the scene. It is capable of gathering one million 3D points at a maximum distance of 100 meters with an accuracy of 5mm. Each 3D point is associated with four values $(x, y, z, l)^T$, where $(x, y, z)^T$ is its Cartesian coordinates in the scanner's local coordinate system, and l is the laser intensity of the returned laser beam.

Each range scan then passes through an automated segmentation algorithm [43] to extract a set of major 3D planes and a set of geometric 3D lines G_i from each scan $i = 1, \ldots, K$. The geometric 3D lines are computed as the intersections of segmented planar regions and as the borders of the segmented planar regions. In addition to the geometric lines G_i, a set of reflectance 3D lines L_i are extracted from each 3D range scan. The range scans are registered in the same coordinate system via the automated 3D-to-3D feature-based range-scan registration method of [9, 44]. The method is based on an automated matching procedure of linear features of overlapping scans. As a result, all range scans are registered with respect to one selected pivot scan. We have also developed a circle-based registration method [10, 8] that we describe in the following sections. The set of registered 3D points from the K scans is called M_{range} (Fig. 1).

3.1. 3D Edge Detection

Each range scan S_i is represented as a 2D array of 3D points $\{\mathbf{P}(k, l), k = 1 \ldots W, l = 1 \ldots H\}$[1]. Two such range images are shown in Fig. 2. Within each range image we consider 3D edges of the following two types: (a) edges caused by surface normal discontinuities (roof edges), and (b) edges caused by depth discontinuities (step edges). Step edges are further divided into edges caused by one surface occluding another (occlusion edges), and edges caused by 3D surface boundaries (boundary edges).

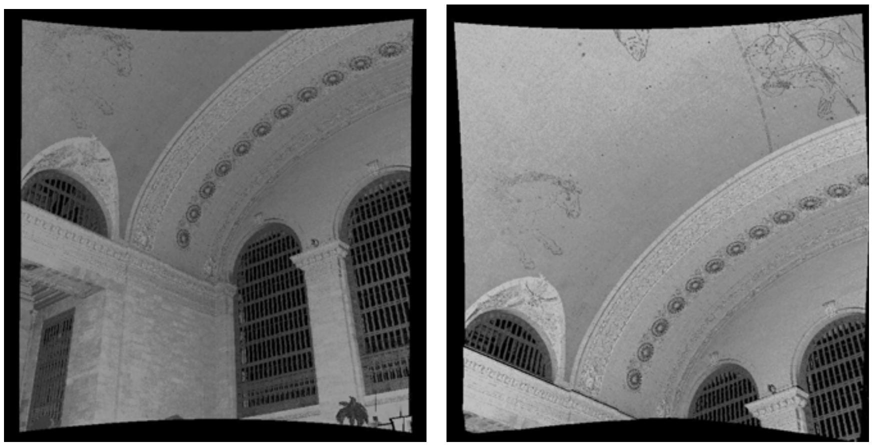

(a) (b)

FIG. 2. Two range images of the interior of the Grand Central Terminal, NYC.

[1]The indices k, l define the position and orientation of the laser-beam which produces the 3D point $\mathbf{P}(k, l)$.

We briefly summarize the algorithm for detecting edges of various types. First the surface orientation change at each point is decomposed into variations along four grid directions. This grid is the 2D structured grid on which each range image is organized (as mentioned in the previous paragraph). We thus obtain four values at every 3D point, that we call directional variation values. In the structured 2D grid we form four 2D images. The intensity value at each pixel is the surface variation (we define it properly in the next paragraphs) of the corresponding 3D point. We call the four 2D images directional variation images. 2D Canny-based edge detection is performed on each image. Finally the 2D edges are combined and projected[2] back to the 3D range image space, providing the final 3D edges due to surface normal discontinuities or depth discontinuities.

The directional variation images are obtained as follows: At each point P, let B_1 and B_2 be its two neighbors along one of the four grid directions (see Fig. 3(e)). The vector from P to B_1 is $\mathbf{V_1}$, and from P to B_2 is $\mathbf{V_2}$. The variation at each direction for point P is defined as $Angle(\mathbf{V_1}, \mathbf{V_2})/\pi$. This provides a value in $(0, 1]$ as the intensity value for this 2D directional variation image.

2D edge detection is performed on each of the four directional variation images. First, Gaussian smoothing is applied to suppress noise. Then, gradients along x and y direction, g_x and g_y, are computed at each pixel using Sobel operators. With g_x and g_y we compute the gradient magnitude g, and determine the edge direction at each point as one the followings: horizontal, vertical, positive diagonal and negative diagonal. Our algorithm then carries out hysteresis thresholding followed by non-maximum suppression to obtain thin continuous edges.

To this point, we have detected all roof and step edges. However, occlusion edges need to be identified and only the foreground edges should be kept in order to reflect the true geometry of the scene (similar to the shadows in 2D images). The earlier step of non-maximum suppression votes off edge points based on magnitude, regardless of whether it is on a foreground surface or a background surface. To find occlusion edges, we map the 2D edge points back to 3D range scan and label a 3D point P if its corresponding pixel p is an edge point. For each edge point P, we check its two neighbors perpendicular to its edge direction. If one of these neighbors is much closer to the scanner and is not an edge point, we mark this neighbor to be a foreground edge point and mark P as non-edge.

Next we remove corner points in order to break the connections between edges of different directions, thereby simplifying edge linking and fitting. Corner points are detected by applying Harris corner detector to every edge point, and testing whether there are more than one principle directions formed by all edge points in its local neighborhood.

The last step is the combination of four edge maps by taking the union of all edge points. From the combined edge map, isolated edge points are deleted, and short gaps (1 or 2 pixels) are filled along the local edge direction. Then continuous edge points are linked by tracing along edge directions. The edge linking utilizes the structured grid on which the range image is represented for resolving neighbors. Only long edges (30 points or more) are being kept for later processing. Fig. 3(f) shows The final combined edge map, in which different colors indicate different edge directions, each detected by one directional variation image.

[2]Each pixel $p(k, l)$ in the grid-point image corresponds to a 3D point $\mathbf{P}(k, l)$.

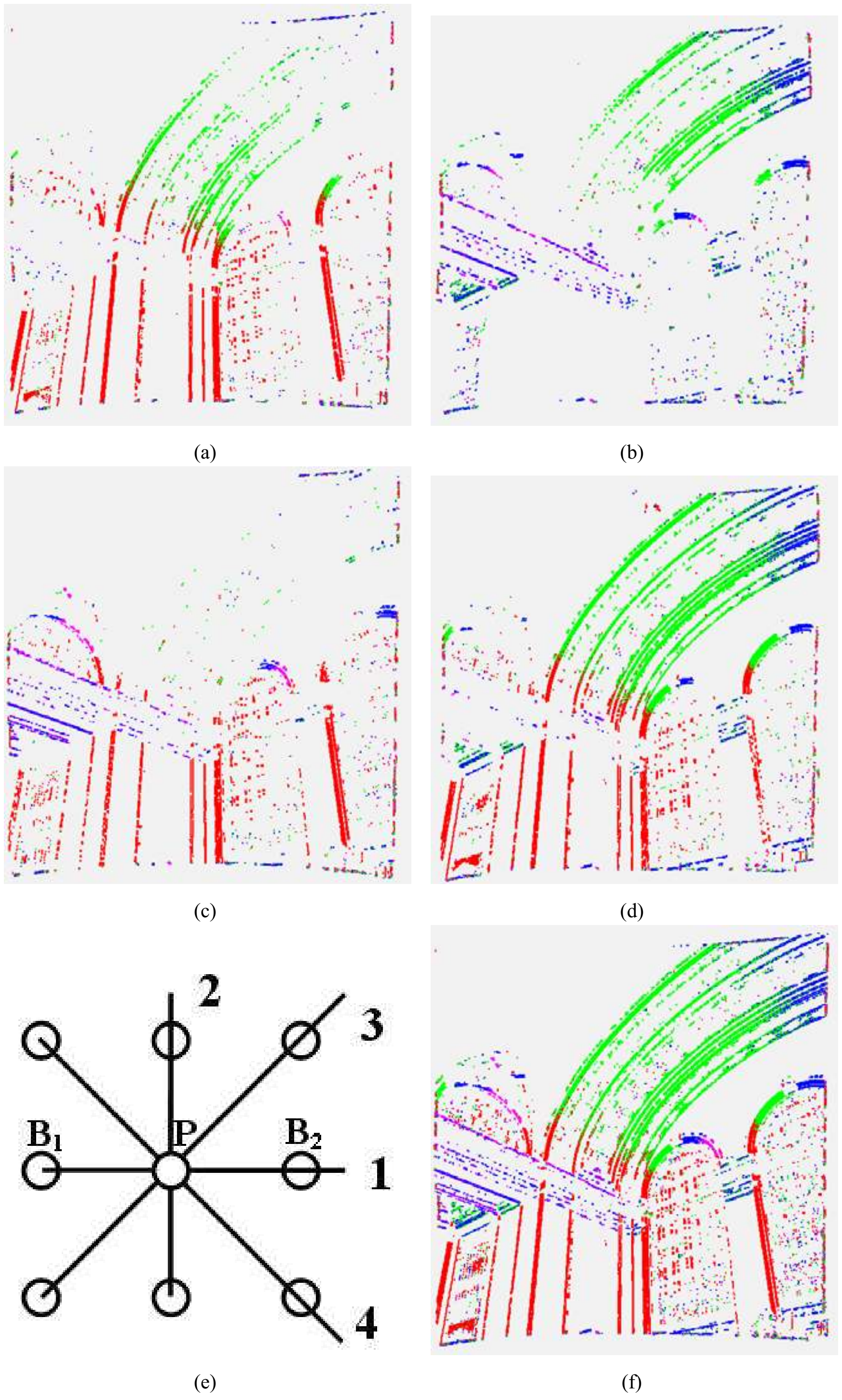


FIG. 3. Edge points of range images of Fig. 2(a). Note that the color at each point (red/green/yellow/magenta) indicates its edge direction (see text), hence the same point usually has the same color in the four edge images. (a)-(d) Edge points detected from Fig. 2(a),(b) respectively. (e) Four grid directions: 1-Horizontal, 2-Vertical, 3-Positive Diagonal, 4-Negative Diagonal. B_1 and B_2 are P's neighbors along direction 1. (f) Combined edge image from (a)-(d).

3.2. 3D Feature Extraction

Our linear feature extraction is described in [9, 44]. In this section we present our algorithms for detecting circular features in 3D space. This is necessary in those cases where linear features are inadequate to register the circular arcs that may be present in the scene. Although the ellipse also exists in our scenes, it is less robust to extract due to its feature of having two focai. Any error or noise in edge point extraction might greatly affect the parameters of the fitted ellipse, and further lead to incorrect computation on transformation.

Each linked set of edges describes a curve in 3D space. For each linked edge from Fig. 3(e), its best-fit line direction $\mathbf{V_{max}}$ and best-fit plane normal $\mathbf{V_{min}}$ are computed. A curve is considered linear if the line fitting error (average distance of all points to the fitted line) is less than a threshold 0.03m (approximate distance between two neighboring 3D range points). For nonlinear curves, the average perpendicular distance of all points to the fitted plane is used to discard 3D curves that are non planar (a generous threshold of 0.5m is used). For each of the remaining planar curves, all points are projected onto their fitted plane. After this process, the 3D curve becomes a set of 2D points in the 2D space Π of the fitted plane. Circle fitting is done in this space.

Taking the common approach of least square fitting[3], we compute the center (a, b) and radius r of the circle by finding an approximate null-vector of a $n \times 4$ design matrix, where n is the number of points on the curve. Consider the circle function $(x-a)^2 + (y-b)^2 - r^2 = 0$. It can be written as $x^2 + y^2 - 2ax - 2by + a^2 + b^2 - r^2 = 0$. Let (x_i, y_i) be the 2D coordinates of all points $p_i (i = 1, ..., n)$ on the curve. Then the circle equation for all points can be expressed as a multiplication of the $n \times 4$ matrix $M = [M_1 \quad M_2 \quad ... \quad M_n]^T$ where $M_i = [x_i^2 + y_i^2 \quad -2x_i \quad -2y_i \quad 1]$ (for $i = 1, ..., n$), with unknown vector $[1 \quad a \quad b \quad a^2 + b^2 - r^2]^T$. The null-vector of the design matrix, computed by SVD, provides the solution. Finally, the circle fitting error is computed as $c_{err} = \sqrt{\frac{\Sigma_i^n (distance(p_i - center) - r)^2}{n}}$. The ratio $(\frac{c_{err}}{r})$ must fall below a threshold (0.02) to verify that the planar 3D curve is a circular arc. Finally, the center of the fitted circle is converted back from Π to the 3D space. We now have three parameters to represent each oriented 3D circle: 3D center point, radius, and plane normal. Fig. 4 shows all the circles with radii between 3.0m and 5.0m. These are the ones most useful for matching in the next step. In the execution, we detect all circles with radii between 2.0m and 20.0m.

3.3. Feature Matching

Our linear feature matching is described in [9, 44]. In this section we present our algorithms for matching circular features in 3D space.

After the oriented 3D circles are extracted from range images, possible matchings between them are hypothesized. The computed transformations are graded using surface consistency[21] and average point distance in the overlapping area between the scans.

Similarity of radii, orientation and relative position between pairs of circles is utilized in the matching phase. In particular, consider a pair of circles (C_1, C_2) from scan S_1 and another pair of circles (C'_1, C'_2) from scan S_2. The pairs would be considered as matching iff

1. Circles C_1, C'_1 have equal radii within a threshold (maximum difference of 0.1m);

[3]A 3D hough transform method will be inefficient, since the radii of circles are unknown and may vary wildly.

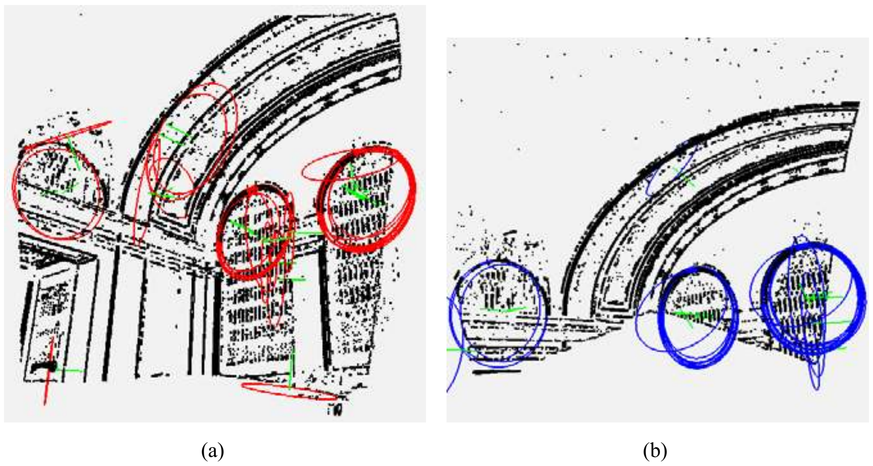


FIG. 4. Circles extracted from the range images of Figs. 2(a) and 2(b), respectively. All edge points are in black, and all fitted circular arcs are represented with colored full circles, with green lines indicating their normals. Note that three circular windows are detected in both images. The images are rotated to the best angle to observe all circles. Therefore some of them appear as ellipses due to the viewing direction.

2. Circles C_2, C_2' have equal radii within a threshold (maximum difference of 0.1m);

3. The distance between the centers of C_1, C_1' equals the distance between the centers of C_2, C_2' within a threshold (maximum difference of 0.2m);

4. The angle between the normals of C_1, C_1' equals the angle between the normals of C_2, C_2' within a threshold (maximum difference of 10^o).

Furthermore, consider a pair of circles (C_1, C_2) from scan S_1 and another pair of circles (C_1', C_2') from scan S_2 that could be considered a valid match according to the previous definitions. A transformation (rotation R followed by a translation T) can be computed by converting the correspondence of a pair of oriented circles to a pair of 3D oriented lines. This approach leads to a robust transformation computation, since it is based on relative position and orientation of the features rather than exact position and orientation of each feature. In particular, two cases are considered:

Case 1: Circles (C_1, C_2) have parallel normals $\mathbf{V_1}$ and $\mathbf{V_2}$ (the same is true for the normals $\mathbf{V_1'}$ and $\mathbf{V_2'}$ of circles (C_1', C_2')) (Fig. 5(a)). Let us consider the oriented line $\mathbf{D}$ that connects the centers of (C_1, C_2) and the oriented line $\mathbf{D'}$ that connects the centers of (C_1', C_2'). If $\mathbf{D}$ is not parallel to $\mathbf{V_1}$ (that means that $\mathbf{D'}$ is not parallel to $\mathbf{V_1'}$), the match of the oriented line $\mathbf{D}$ with $\mathbf{D'}$ and $\mathbf{V_1}$ with $\mathbf{V_1'}$ can provide a reliable transformation (closed form formula [44]). Otherwise ($\mathbf{D}$ is parallel to $\mathbf{V_1}$) a reliable transformation can not be computed. Note that if the length of $\mathbf{D}$ (or $\mathbf{D'}$) is below an empirically determined threshold (it is 5m for the Grand Central Terminal dataset) the above computation is not performed (that means that the candidate match is discarded). This effectively improves the performance by not considering pairs of spatially close features[4].

Case 2: Circles (C_1, C_2) do not have parallel normals $\mathbf{V_1}$ and $\mathbf{V_2}$ (the same is true for the normals $\mathbf{V'_1}$ and $\mathbf{V'_2}$ of circles (C'_1, C'_2)) (Fig. 5(b)). Then, the two pairs of oriented lines $(\mathbf{V_1}, \mathbf{V_2})$ and $(\mathbf{V'_1}, \mathbf{V'_2})$ are used for the computation of a reliable transformation (closed form formula [44]).

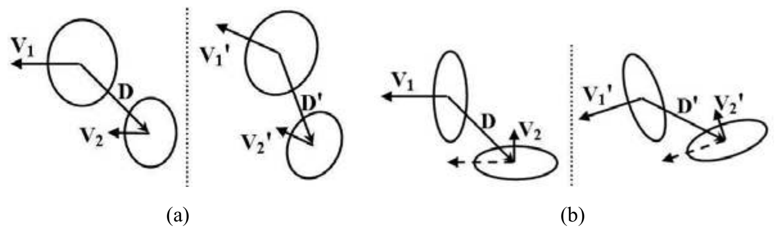


FIG. 5. Two cases of matching circle pairs. The dash line separates scan S_1 from S_2. The radii and relative position of the two circles from S_1 must be similar to those from S_2. (a) Case 1: two circles have parallel normals. V_1, D and $V'_1, D'$are used to compute transformation. (b) Case 2: two circle normals are not parallel. V_1, V_2 and $V'_1, V'_2$are used to compute transformation.

From each valid matching circle pairs, a candidate transformation is computed as described above. Each transformation is verified for correctness as follows. Based on the fact that overlapping images are captured from nearby positions, we discard all rotation matrices with diagonal elements smaller than 0.7 (allowing 45^o tilting of the range scanner about each of its $x/y/z$ axes). Note that this step reduces the number of possible transformations and thus speeds up the algorithm, but is not otherwise necessary. Then we test whether the transformation causes surface inconsistency [21]. Finally, from all verified transformations, the one achieving the smallest average distance between overlapping range points is chosen as the best.[5]

3.4. Experimental Results

Our automated method is used for registration of the interior scans of Grand Central Terminal in NYC (a large-scale landmark urban structure). The best transformation of the two corner scans of Fig. 2 provides a registration error (average point distance in the 55.7% overlapping area) of 0.95cm. Within a few iterations of ICP an optimal transformation with a registration error of 0.90cm is obtained (Fig. 6).

Also, we registered other scans of this hall with the same technique. The entire hall is roughly in rectangular shape with an arched ceiling. Fig. 7 shows a few typical scans on the front wall ((a)(c)) and the side wall ((e)(g)), together with circles extracted from them. Note that the lower parts of the walls (e.g.(c)(g)) contain lines and planes, and are therefore registered with our linear-feature based technique [9, 44]. The upper regions with

[4]Our segmentation, as seen in Fig. 4, produces many similar spatially close circles. We decided not to average them in order not to decrease accuracy.

[5]Note that an approach similar to association graphs [1] would generate a very large search space.

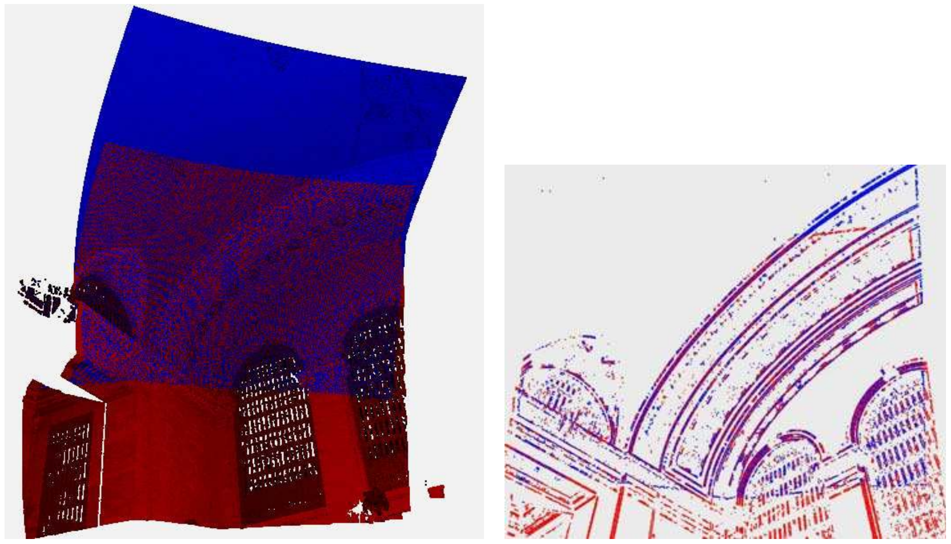


FIG. 6. Registered images of Figs. 2(a) and 2(b). They are colored to highlight overlapping area. (a) All image points. (b) Edge points at overlapping area.

very few linear features, e.g.(a)(e), are registered with their lower neighboring scans (c)(g) respectively, by matching overlapping circular windows.

In Fig. 9, registered edge points from 23 scans are visualized. There are another 14 scans not shown for clarity of presentation. Among all 37 scans, 20 of them are lower parts registered with lines, 13 of them are the upper parts registered with their lower neighbor scans based on overlapping circular windows. Three scans are manually registered, because they are cylindrical ceiling patches without any distinguishing geometric shape information. In Table 1 we report the performance of 13 registrations based on circles. When registering the last two pairs, a long execution time is experienced due to a large number of valid transforms from the precisely extracted circles around the window frame (as in Fig. 7(b)(d)). To avoid unnecessary computations, we set the program to terminate when the average distance falls below 0.03cm (approximate distance between two neighboring points). The values in columns *RT*, *Dist_match*, *Time* are therefore recorded up to the point when an accurate enough result is reached. In Table 2 we report the performance of line-based registration [9, 44] in the lower part of the building. In Fig. 8, more results are shown, including registered scenes, edge points, and part of a 3D model constructed using the registered range points.

4. 2D-TO-3D IMAGE-TO-RANGE REGISTRATION

We present our automated 2D-to-3D image-to-range registration method used for the automated calibration and registration of a single 2D image I_n with the 3D range model M_{range}. The computation of the rotational transformation between I_n and M_{range} is achieved by matching at least two vanishing points computed from I_n with major scene directions computed from clustering the linear features extracted from M_{range}. The method

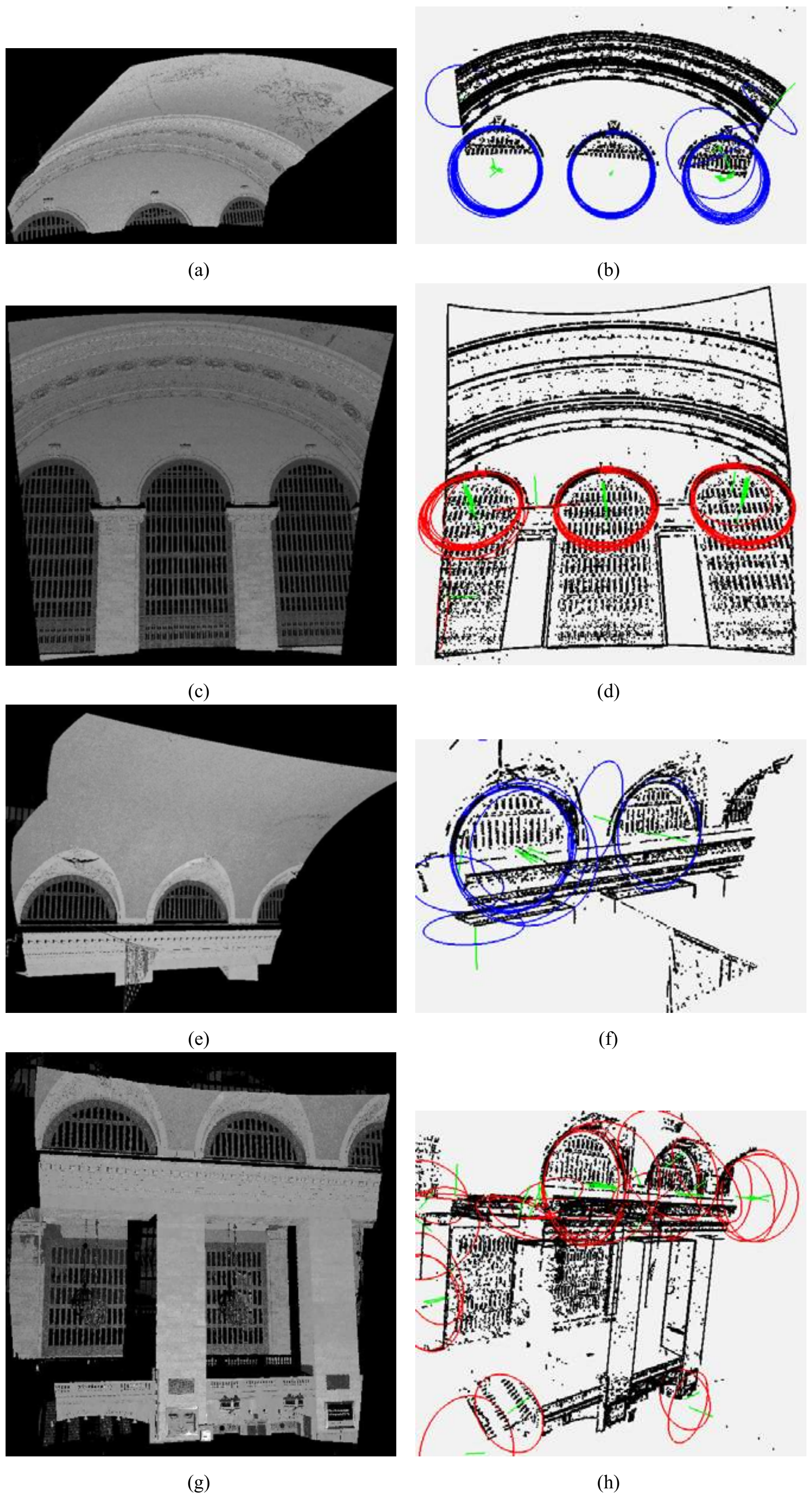


FIG. 7. Four side wall scans (left column), extracted edges and fitted circles (right column).

TABLE 1
Experimental results of circle-based 3D-to-3D registration. Grand Central Terminal dataset. Columns: Number of circles in images ({left, right}); Number of candidate transformations; Average point distance assuming best transformation after circle-matching; Average point distance after ICP optimization; Percentage of overlap; Execution time, including circle extraction and matching (on a Linux-based 2GHz Xeon-Processor with 2GB of RAM).

Circles	RT	Dist_match	Dist_icp	Overlap	Time
{26, 24}	544	0.95 cm	0.90 cm	55.7%	6 min
{24, 39}	980	1.11 cm	1.00 cm	29.1%	9 min
{21, 39}	15	3.42 cm	1.28 cm	16.1%	1 min
{24, 20}	748	2.13 cm	0.77 cm	38.4%	7 min
{13, 30}	126	2.01 cm	0.84 cm	28.9%	2 min
{21, 26}	534	1.68 cm	0.90 cm	35.5%	6 min
{23, 11}	29	4.26 cm	0.87 cm	28.7%	1 min
{14, 31}	18	2.65 cm	0.93 cm	27.8%	2 min
{31, 13}	58	2.34 cm	0.98 cm	23.0%	2 min
{37, 26}	67	3.83 cm	0.87 cm	37.2%	2 min
{23, 35}	310	1.20 cm	0.84 cm	26.7%	7 min
{49, 41}	3054	2.81 cm	1.02 cm	38.7%	58 min
{50, 38}	931	1.83 cm	0.92 cm	44.6%	10 min

TABLE 2
Experimental results of line-based 3D-to-3D registration [9, 44]. Grand Central Terminal dataset. Columns: Number of lines in images ({left, right}); Number of matching line-pairs before ICP; Average point distance assuming best transformation from line-based matches; Number of matching line-pairs after ICP; Average point distance after ICP optimization; Execution time, for line matching (on a Linux-based 2GHz Xeon-Processor with 2GB of RAM).

Line Pairs	N	Dist_match	N	Dist_icp	Time
{318, 214}	5	2.31 cm	8	0.934 cm	1 sec
{318, 261}	19	2.16 cm	21	1.02 cm	10 sec
{261, 239}	13	1.51 cm	13	1.14 cm	10 sec
{239, 236}	22	1.05 cm	22	0.988 cm	10 sec
{241, 230}	3	2.26 cm	2	1.30 cm	9 sec
{241, 230}	3	1.53 cm	3	1.33 cm	10 sec
{230, 175}	1	2.66 cm	2	1.49 cm	11 sec
{175, 175}	16	1.05 cm	17	0.932 cm	10 sec
{175, 247}	2	3.32 cm	4	0.981 cm	9 sec
{262, 175}	2	3.37 cm	4	1.39 cm	10 sec
{247, 262}	5	2.97 cm	10	1.04 cm	9 sec
{262, 275}	16	2.00 cm	18	1.04 cm	10 sec
{275, 254}	11	1.15 cm	11	1.08 cm	10 sec
{194, 262}	2	3.79 cm	8	1.10 cm	11 sec
{198, 239}	10	3.13 cm	19	1.39 cm	12 sec
{253, 230}	2	2.26 cm	2	1.19 cm	12 sec

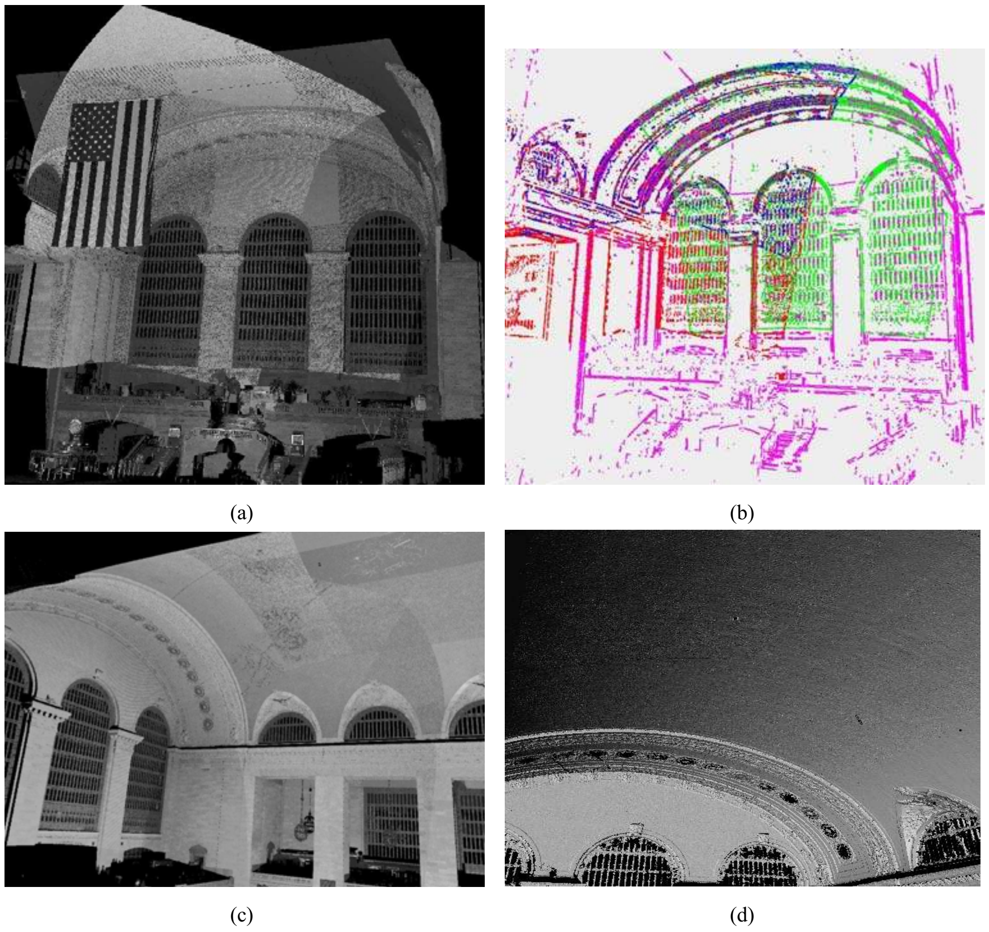


FIG. 8. Registration results. (a) Four out of the 37 automatically registered scans shown for clarity. (b) Edge points of (a). Four colors represent edge points from four scans. (c) Eight out of the 37 automatically registered scans shown for clarity. (d) 3D mesh model generated by the Ball Pivoting Algorithm [4]. The smooth ceiling implies the registration is tight and seamless.

is based on the assumption that the 3D scene contains a cluster of vertical and horizontal lines. This is a valid assumption in urban scene settings.

The internal camera parameters consist of focal length, principal point, and other parameters in the camera calibration matrix $\mathcal{K}$ [17]. They are derived from the scene's vanishing points, whereby the 2D images are assumed to be free of distortion after we remove lens deformations, as described in [30]. Finally, the translation between I_n and M_{range} is computed after 2D and 3D lines from the 2D image and 3D model and are extracted and automatically matched. With this method, a few 2D images can be independently registered with the model M_{range}. The algorithm will fail to produce satisfactory results in parts of the scene where there is a lack of 2D and 3D features for matching. Also, since each 2D image is independently registered with the 3D model, valuable information that can be extracted from relationships between the 2D images (SFM) is not utilized. In order to

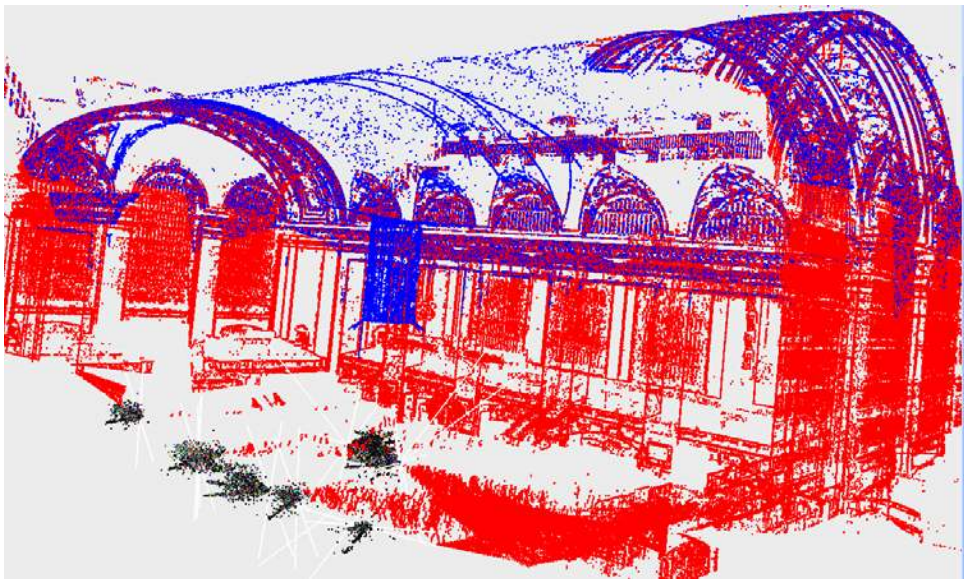


FIG. 9. Registered edges of 23 scans, with color indicating scans from upper parts (blue) and lower parts (red). The other 14 scans are not displayed for clarity of presentation; these scans compose a side wall closer to our point of view and symmetric to the wall being displayed. Tables 1 & 2 report the registration performance for the upper (circle-based registration) and lower (line-based registration) parts.

solve the aforementioned problems, an SFM module (Sec. 5) and final alignment module (Sec. 6) has been added into the system. These two modules increase the robustness of the reconstructed model, and improve the accuracy of the final texture mapping results. Therefore, the 2D-to-3D image-to-range registration algorithm is used in order to register a few 2D images (five shown in Fig. 1) that produce results of high quality. The final registration of the 2D image sequence with the range model M_{range} is performed after SFM is utilized (Sec. 5).

In this section, we present a system that can automatically register 2D images with 3D range data at interactive rates. Our contributions with respect to 2D-to-3D registration can be summarized as follows:

• We have developed a working system that is able to independently register 2D images to 3D models at interactive rates. This system requires minimal user interaction. Note that after a few 2D images are registered to the 3D model the multiview geometry approach (Sec. 5) is utilized for registering all images with the 3D range model.

• The whole space of possible matches between 3D and 2D linear features is explored efficiently (unlike probabilistic methods like [42]). That improves the possibility of convergence of our algorithm.

• Earlier systems ([29, 42]) require the extraction of major facades, rectangles, or other higher-order structures from the 2D and 3D datasets. Our new method, on the other hand, utilizes 3D and 2D linear features for matching without significant grouping. This increases the generality of our algorithm since we make fewer assumptions about the 3D scene. Scenes with various layers of planar facades, or without clear major facades can thus be handled.

• This paper's method utilizes vanishing points and major 3D directions, but it does not require them to be orthogonal as most earlier methods assume.

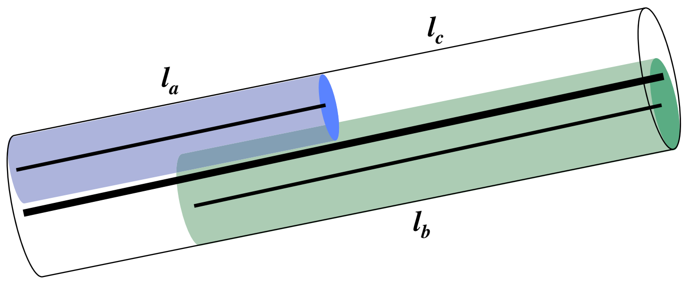
4.1. 3D Feature Extraction for 2D-to-3D Registration

The range-image acquisition process and segmentation is described in Sec. 3. As we described in that section, all range scans are registered with respect to one selected pivot scan, in the scene's coordinate system. The 3D line extraction step is based on the segmentation method of [43], whereas the major directions clustering is based on [29]. (Note that if 3D information is provided in terms of a CAD model, then the 3D line extraction step is trivial.) The result of this process is a set of line clusters $\mathcal{L}^{3D}$. Each line in a cluster has similar orientation as every other line in the same cluster. The set of line clusters are then sorted based on the number of lines in each cluster. We do not assume knowledge of vertical or horizontal directions for the line clusters as in our previous method [29]. Each 3D line is thus associated with a cluster id, e.g. for the 3D lines in cluster $\mathcal{L}_i^{3D}$, their cluster id is i. In the next step, 3D features are extracted. First, an initial user-defined radius (e.g. 0.1m) is assigned to each 3D line. Then, a line merging step generates the final 3D features. This reduces the number of features, and thus increases the efficiency of the matching stage (Sec. 4.3). In this step, each pair of 3D lines (l_a, l_b) with the same cluster id are merged into a new line l_c (Fig. 10) iff a) the distance between them are smaller than the sum of their radii, and b) their projections on l_c overlap. The merging procedure is continued until there are no two remaining 3D lines that can be merged. The final result is a set of 3D lines, each of which is associated with a cluster id and radius.

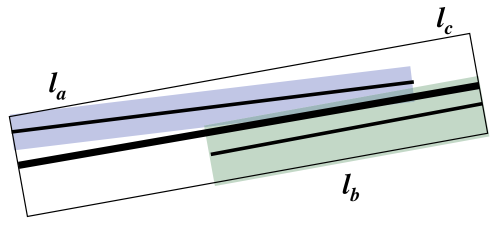
4.2. 2D Feature Extraction, Internal Camera Calibration, & Rotation Computation

The internal camera parameters (focal length and principal point) of the camera sensor can be calculated from one 2D image if the image contains at least two vanishing points (i.e. the 3D scene that the image is viewing has at least two major scene directions). We use our previously developed robust methods to generate and cluster 2D lines from a 2D image [42]. The result is a set of major vanishing points $\mathbf{VP} = \{\mathbf{VP_1}, \mathbf{VP_2}, \cdots, \mathbf{VP_n}\}$. Using the methods described in [42] we can compute the center of projection $\mathbf{COP} = [C_x, C_y, C_z]^T$ (effective focal length and principal point expressed in pixels) by utilizing **three** orthogonal vanishing points. In [29] we described an iterative method for estimating the internal calibration parameters from two orthogonal vanishing points.

In this section we present an additional method [28] for the calculation of the effective focal length f and of the rotation $\mathbf{R}$. We are using two vanishing points and two major 3D directions. We, however, do not assume that these directions are orthogonal to each other. Orthogonality is prominent in urban scenes, but is not always present. Our method starts with an initial estimate f_{init} of the effective focal length, and of the principal point $\mathbf{P}_{init}$. f_{init} is included in the Exif meta-data, information that is now provided by most digital cameras. $\mathbf{P}_{init}$ is estimated by the center of the image. Based on these estimates an initial center of projection $\mathbf{C}_{init}$ is determined. This is the origin of the camera coordinate system (Fig. 11). Let us consider a vanishing point $\mathbf{V}_i$ extracted from a 2D image. The 3D coordinates of $\mathbf{V}_i$ in the camera coordinate system are $[(V_i)_x - (P_{init})_x, (V_i)_y - (P_{init})_y, f_{init}]^T$.



3D feature merging (l_a and l_b merged into l_c)



2D feature merging (l_a and l_b merged into l_c)

FIG. 10. Example of new type of 3D and 2D features and their merging steps.

Thus, the normalized vector $\mathbf{D}_i^{2D} = u(\mathbf{C}_{init}\mathbf{V}_i)$[6] represents the 3D direction that generates the vanishing point $\mathbf{V}_i$. This direction is expressed in the camera coordinate system. Our goal is to match each vanishing point with its corresponding 3D direction extracted by the 3D range model (see Sec. 4.1). This correspondence leads to the calculation of the focal length and of the rotation $\mathbf{R}$. Let us represent each 3D line cluster in $\mathcal{L}^{3D}$ (Sec. 4.1) by its 3D direction $\mathbf{D}_j^{3D}, j = 1 \ldots n$ (where n is the number of extracted 3D clusters).

The next step is to find the matching pairs of directions $< \mathbf{D}_i^{2D}, \mathbf{D}_j^{3D} >$. Consider for the moment that we know the correspondence between vanishing points (expressed in the camera coordinate system) and 3D directions (expressed in the world coordinate system). It is known that with the principal point fixed at the center of image, two pairs ($< \mathbf{D}_a^{2D}, \mathbf{D}_a^{3D} >, < \mathbf{D}_b^{2D}, \mathbf{D}_b^{3D} >$) of matching vanishing point/3D directions are enough for the computation of the focal length f. The focal length f (which is $|\mathbf{CP}|$ in Fig. 11) can be computed via the following equations (triangles $\mathbf{CV_aP}$, $\mathbf{CV_bP}$ and $\mathbf{CV_aV_b}$):

$$|\mathbf{CV}_a|^2 = |\mathbf{PV}_a|^2 + f^2$$

$$|\mathbf{CV}_b|^2 = |\mathbf{PV}_b|^2 + f^2$$

$$|\mathbf{V}_a\mathbf{V}_b|^2 = |\mathbf{CV}_a|^2 + |\mathbf{CV}_b|^2 - 2 \cdot |\mathbf{CV}_a| \cdot |\mathbf{CV}_b| \cdot \cos\alpha$$

where α is the angle between $\mathbf{D}_a^{3D}$ and $\mathbf{D}_b^{3D}$. (Note that the vanishing points $\mathbf{V_a}$ and $\mathbf{V_b}$ have been computed by using the initial estimates f_{init} and $\mathbf{P}_{init}$. The above computation

[6]We use the notation $u(\mathbf{v})$ for describing the unit vector derived from $\mathbf{v}$.

leads to the calculation of a focal length that conforms to the 3D directions $\mathbf{D}_a^{3D}$ and $\mathbf{D}_b^{3D}$.) From the above equations, we can get a quartic equation:

$$a \cdot f^4 + b \cdot f^2 + c = 0$$

where $a = \sin^2 \alpha$, $b = \sin^2 \alpha(|\mathbf{PV}_a|^2 + |\mathbf{PV}_b|^2) - |\mathbf{V}_a\mathbf{V}_b|^2$, $c = (\frac{|\mathbf{V}_a\mathbf{V}_b|^2 - |\mathbf{PV}_a|^2 - |\mathbf{PV}_b|^2}{2})^2 - \cos^2 \alpha |\mathbf{PV}_a|^2 |\mathbf{PV}_b|^2$. Solving this equation, one obtains the refined focal length: $f = \sqrt{\frac{\sqrt{b^2-4ac}-b}{2a}}$. Since $\mathbf{D}_a^{3D} \neq \mathbf{D}_b^{3D}$, $\sin \alpha$ will never be equal to 0. Finally, the rotation $\mathbf{R}$ is computed based on these two pairs of matching directions [13].

Based on the above analysis, the task of our system is to find two matching pairs of vanishing point/3D directions. Intuitively, pairs ($< \mathbf{D}_a^{2D}, \mathbf{D}_a^{3D} >$, $< \mathbf{D}_b^{2D}, \mathbf{D}_b^{3D} >$) for which the angle between $\mathbf{D}_a^{2D}$ and $\mathbf{D}_b^{2D}$ is not similar to the angle between $\mathbf{D}_a^{3D}$ and $\mathbf{D}_b^{3D}$ can be rejected. As a result, we have a list of matching candidates, each of which contains two pairs of matching vanishing points and 3D directions, a refined focal length and a rotation. For each one of these candidates we can apply the algorithm described in the next section for calculating the camera position, and finally keep the result that provides the maximal alignment between the 2D image and 3D model.

In the worst case scenario though all pairs of directions have similar angles (this scenario is easily realizable in urban scenes where most angles between major directions is 90 degrees). In this case there are $\binom{n}{2}\binom{m}{2}$ candidate matching pairs of directions (where n is the number of 3D and m the number of vanishing points). Even though this is not a large search space (n and m are small in most urban scenes), testing all hypotheses involves the computation of the translation (see Sec. 4.3). This is computationally inefficient for the purposes of an interactive system, where a response time of up to 10 seconds per image is appropriate. For these reasons we let the user to implicitly provide the correct pair of matching directions, by rotating the 3D model to an orientation that produces a rendering that is similar (but not exactly the same) to the real 2D image. As shown in Figs. 15(b) and 16(c), the rotated 3D view (left) is similar (but not exactly the same) to the 2D image (right). This user-assisted rotation can approximately align the corresponding 2D and 3D directions.

The aforementioned user interaction not only increases the computational efficiency of the whole system, but also makes the registration problem tractable. In general, without constraining the possible locations of 2D cameras wrt the 3D model, the 2D-to-3D registration problem becomes intractable. This is due to the existence of a possible large set of solutions. For example, a photograph of one of the columns of the 3D structure of Fig. 16 can be matched with any of the symmetric 3D columns of the real scene. By selecting a synthetic view that is similar, but not exactly the same as the 2D image, the user can provide an approximate field of view to help the matching algorithm. In particular, only 3D features that are viewable in the synthetic 3D view are used for matching 2D image features. Note here that all earlier approaches still require implicit user interaction in order to assist in that direction. For example in our earlier work [29] the user needs to explicitly provide the match between vanishing points and 3D directions. In that earlier system, the user also needs to match facades between the 2D image and 3D model. The approach presented in this section is more natural and leads to faster interaction time.

4.3. Translation Computation

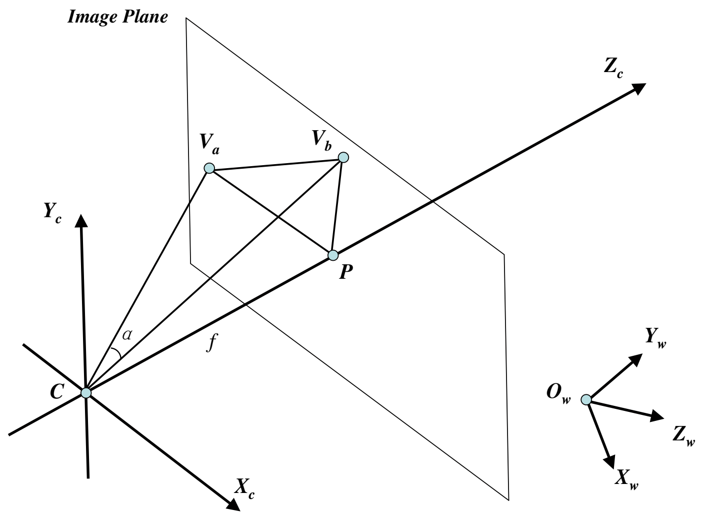


FIG. 11. Rotation and focal length computation based on two vanishing points and their corresponding 3D directions (not shown in this image).

In this section we present the algorithm that automatically computes the translation between one 3D range scan and one 2D image (Fig. 13).

As described in section 4.2, a list of matching candidates, named $\mathcal{M}$, is obtained. Each element in $\mathcal{M}$ contains a matching pair of two vanishing points and two 3D directions, a refined focal length and a rotation. In this section, a 2D camera position will be computed for each candidate in $\mathcal{M}$. Our method of finding the camera position follows a hypothesis-and-test scheme by matching the extracted 3D and 2D features based on our original framework [29]. A number of major differences with the aforementioned method make our algorithm more general and more robust. In particular, our current algorithm does not require the extraction of planar facades, and does not require the grouping of low-level features in higher-order structures. Scenes that do not pertain clear major facades (such as the example of Figs. 16, where various layers of planar facades exist) can now be successfully handled. Also since all low-level features are used without significant grouping, more robust results are achieved. We now present a detailed description of our algorithm. First, a candidate from $\mathcal{M}_i$ is selected, i.e. the matching pair of vanishing points and 3D directions are $< \mathbf{V}_a, \mathbf{V}_b >$ and $< \mathbf{D}_a^{3D}, \mathbf{D}_b^{3D} >$; the refined focal length is f_i and the rotation is $\mathbf{R}_i$. The camera position (translation) is then computed in the following six steps (Fig. 13):

Step 1 A hypothetical match between two pairs of 3D and 2D lines is selected (the algorithm will go over all possible such selections). Let us call these pairs $< l_a^{3D}, l_a^{2D} >$ and $< l_b^{3D}, l_b^{2D} >$ (l_a^{3D} and l_b^{3D} are 3D lines extracted from the 3D model, and l_a^{2D} and l_b^{2D} 2D lines extracted from the 2D image).

Step 2 [Computation of camera position in world coordinate system (translation) based on the match of l_a^{3D} with l_a^{2D}] As shown in Fig. 12, **A** and **B** are the endpoints of l_a^{3D} and

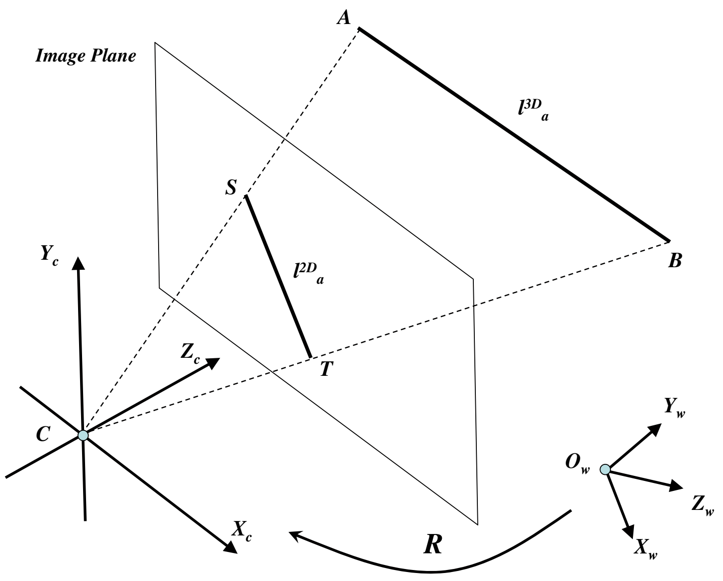


FIG. 12. Camera position computation based on a match between 3D feature $\mathbf{AB}$ with image feature $\mathbf{ST}$.

S and **T** are the endpoints of l_a^{2D}. **C** is the center of projection. If l_a^{3D} matches exactly with l_a^{2D}, then in the camera coordinate system, **C, S** and **A** should be collinear. The same applies for **C, T** and **B**. We thus consider **C** as the intersection point of the following two lines: a) one that passes through **A** having the orientation of line $\mathbf{CS}$ and b) one that passes through **B** having the orientation of line $\mathbf{CT}$. To compute the world coordinates of **C**, we need to know the orientations of $\mathbf{CS}$ and $\mathbf{CT}$ in the world coordinate system. We know, however, the orientations of $\mathbf{CS}$ and $\mathbf{CT}$ in the camera coordinate coordinate system, say $\mathbf{n}_a$ and $\mathbf{n}_b$. We have also computed the rotation R which brings the camera and world coordinate systems into alignment (see previous section). We can thus compute the orientations of $\mathbf{CS}$ and $\mathbf{CT}$ in the world coordinate system as: $R \cdot \mathbf{n}_a$ and $R \cdot \mathbf{n}_b$. Then, the camera position is obtained by finding the intersection of two 3D lines: a) one of which passes through **A** with the orientation of $R \cdot \mathbf{n}_a$ and b) one which passes through **B** with the orientation of $R \cdot \mathbf{n}_b$ [7]. Finally, this computed center of projection is used to project l_b^{3D} onto the image plane. If the projection of l_b^{3D} overlaps with l_b^{2D} (within a threshold of 80%), then the camera position computed using (l_a^{3D}, l_a^{2D}) is verified by the pair (l_b^{3D}, l_b^{2D}). We therefore move to the next step. Otherwise, we return to step 1 (i.e. the match is discarded) to pick another set of hypothetical matching lines.

Step 3 Step 2 is repeated assuming as hypothesis the match between l_b^{3D} and l_b^{2D}. The newly computed center of projection is used to compute the overlap between l_a^{2D} and the projection of l_a^{3D}. If this overlap is less than a threshold (i.e. the computed **C** is not verified

[7] **A** and **B** are both expressed in the world coordinate system

by (l_a^{3D}, l_a^{2D}), we return to step 1 (i.e. the match is discarded). Otherwise, we proceed to the next step.

Step 4 Step 2 has thus computed a camera position $\mathbf{C_1}$ by the hypothesis (l_a^{3D}, l_a^{2D}) [verified by (l_b^{3D}, l_b^{2D})], while step 3 has computed a camera position $\mathbf{C_2}$ by the hypothesis (l_b^{3D}, l_b^{2D}) [verified by (l_a^{3D}, l_a^{2D})]. In this step, the weighted average (based on the amount of overlap) of these two camera positions is computed and saved in a list $\mathcal{T}$.

Step 5 Steps 1 to 4 are repeated for all possible pairs of pairs of 3D and 2D lines ($< l_a^{3D}, l_a^{2D} >, < l_b^{3D}, l_b^{2D} >$). All verified camera positions (see Step 4) are stored in a list $\mathcal{T}$. Then, for each position in $\mathcal{T}$, all 3D lines are projected onto the image plane. For each of the projected 3D lines, a possible matching 2D line is found by searching around its projection. This region is bounded by the radius of the 3D and 2D lines. The number of found matches grades this camera position. If the grade of a camera position is less than a threshold, it is removed from the list $\mathcal{T}$.

Step 6 The remaining camera positions in $\mathcal{T}$ are optimized by two steps. First, for each camera position $\mathbf{C}_i$ a refined position is found. This is achieved by searching around a small neighborhood of $\mathbf{C}_i$ in order to maximize the overlap between the matching 3D and 2D lines. Then this refined position is further optimized by an iterative algorithm. In each iteration, the current camera position is used to generate a list of matching 2D and 3D lines from the whole 2D and 3D feature space. A new camera position is found by minimizing the error distance between the 2D lines and the projections of their matching 3D lines. The algorithm converges at the point when the error distance remains constant. The camera position computed after the two optimization steps are the final result.

The camera position in $\mathcal{T}$ with the maximum grade is picked as the best one for the matching candidate $\mathcal{M}_i$. This is normally correct, but the list is still kept as well in case that the one with the maximum grade is not the best. Then, the user can select other positions in the list. This maximum grade is also used as the grade for $\mathcal{M}_i$. For each matching candidate in $\mathcal{M}$, a list of camera positions is computed by these 6 steps and a grade is assigned. Then, the list $\mathcal{M}$ is sorted based on the grade and the one with the maximum grade is selected as the best one but the user also can select other results in $\mathcal{M}$.

4.4. Results & Conclusions

We are presenting results from real experiments in three urban settings that we name 1 (Fig. 15), 2 (Fig. 14), and 3 (Fig. 16). Buildings 1 and 2 are the exteriors of regular urban structures. Building 3 is the interior of Grand Central Station, a scene of architectural complexity and beauty. The 3D range model for all buildings was generated by our group. First a number of 3D range scans of each structure was acquired using a Leica HDS 2500 time-of-flight laser range scanner [27]. This scanner provides absolute 3D range measurements up to a distance of 100 meters, and at an accuracy of 6mm. Each 3D point is associated with reflectance information, that corresponds to the amount of laser-intensity getting back to the range sensor[8]. We then segment each range scan, extract linear 3D features, and register the scans in a common coordinate system. Figs. 14, 15, and 16 provide individual registration results, as described in our technical sections. Note than in the case of 15(b) and 16(c) the user needs to orient the 3D range model in a position that simulates the 2D

[8]Note that the reflectance depends on various parameters (distance, orientation and surface material) and is not the actual color of the object as captured by a 2D digital camera.

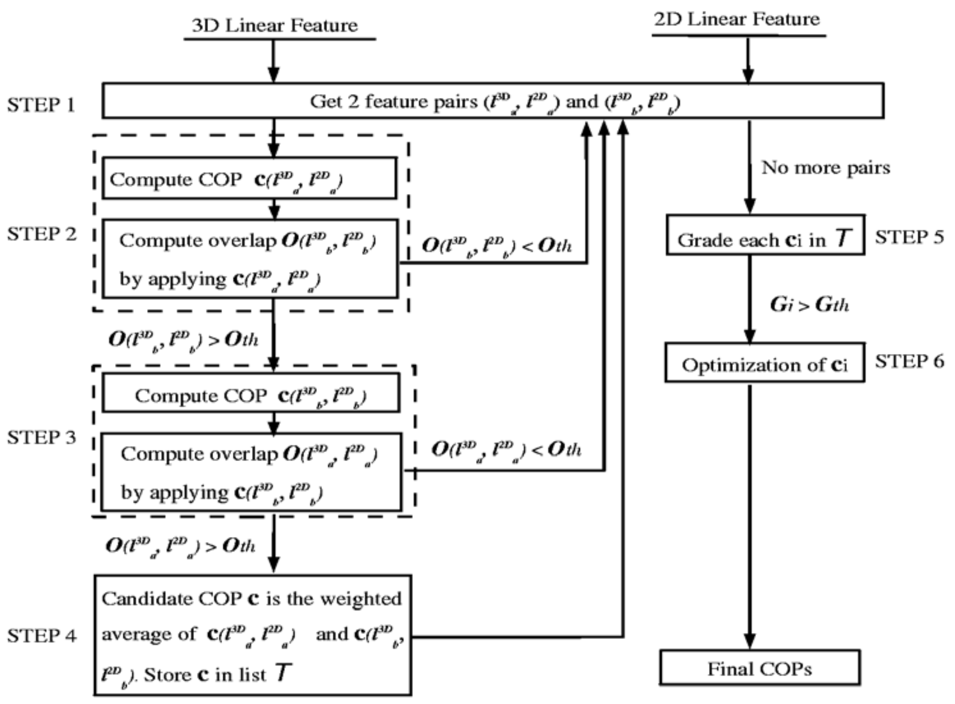

3D Linear Feature
2D Linear Feature
STEP 1
Get 2 feature pairs (l^{3D}_a, l^{2D}_a) and (l^{3D}_b, l^{2D}_b)
Compute COP $\mathbf{c}(l^{3D}_a, l^{2D}_a)$
STEP 2
Compute overlap $O(l^{3D}_b, l^{2D}_b)$ by applying $\mathbf{c}(l^{3D}_a, l^{2D}_a)$
$O(l^{3D}_b, l^{2D}_b) < Oth$
$O(l^{3D}_b, l^{2D}_b) > Oth$
Compute COP $\mathbf{c}(l^{3D}_b, l^{2D}_b)$
STEP 3
Compute overlap $O(l^{3D}_a, l^{2D}_a)$ by applying $\mathbf{c}(l^{3D}_b, l^{2D}_b)$
$O(l^{3D}_a, l^{2D}_a) < Oth$
$O(l^{3D}_a, l^{2D}_a) > Oth$
STEP 4
Candidate COP $\mathbf{c}$ is the weighted average of $\mathbf{c}(l^{3D}_a, l^{2D}_a)$ and $\mathbf{c}(l^{3D}_b, l^{2D}_b)$. Store $\mathbf{c}$ in list T
No more pairs
Grade each $\mathbf{c}$i in T
STEP 5
$Gi > Gth$
Optimization of $\mathbf{c}$i
STEP 6
Final COPs

FIG. 13. Camera position (translation) computation flowchart [COP stands for camera position (or center of projection) in this flowchart]. Through step 1 all possible pairs of matched 3D and 2D lines ($< l_a^{3D}, l_a^{2D} >$ and $< l_b^{3D}, l_b^{2D} >$) are selected (l_a^{3D} and l_b^{3D} are 3D lines extracted from the 3D model, and l_a^{2D} and l_b^{2D} 2D lines extracted from the 2D image). Step 2 computes a camera position based on $< l_a^{3D}, l_a^{2D} >$. The pair $< l_b^{3D}, l_b^{2D} >$ is used for the verification of this position. If the overlap between l_b^{2D} and the projection of l_b^{3D} on the image is smaller than O_{th} (20%) (i.e. the position is not verified) a new pair is selected (step 1). Otherwise a similar computation is carried out for the pair $< l_b^{3D}, l_b^{2D} >$ (step 3). If steps 2 and 3 produce two verifiable camera positions, a weighted average is computed (step 4). This average represents the position that is generated by the hypothesis ($< l_a^{3D}, l_a^{2D} >$ and $< l_b^{3D}, l_b^{2D} >$). All verified camera positions are stored in a list $\mathcal{T}$. After all pairs have been explored, each position in $\mathcal{T}$ is graded by projecting all 3D lines on the 2D image space (step 5). Positions with high grade (greater than G_{th} number of matches) survive to the final optimization step 6.

color image. As you can see from these figures this simulation need not be exact. It is necessary for assistance in matching vanishing points with 3D directions (Sec. 4.2) in order for our system to perform in interactive rates (5-10 seconds for matching per image). Table 3 presents quantitative results for successful automated registrations (see caption for more details). A 3-5 pixel distance between the matched 2D and projected 3D lines is due to noise in the line extraction process. Our texture-map results are of extremely high quality though. Out of 18 total images tried for building 1, 13 were registered successfully, whereas 5 have slight errors. Out of 8 total images tried for building 2, 7 were registered successfully, whereas 1 has slight errors. Finally, out of 10 total images tried for building 3, 6 were registered successfully, whereas 4 have slight errors. In all cases the first step (Sec. 4.2) never fails since these scenes contain at least two vanishing points. The second

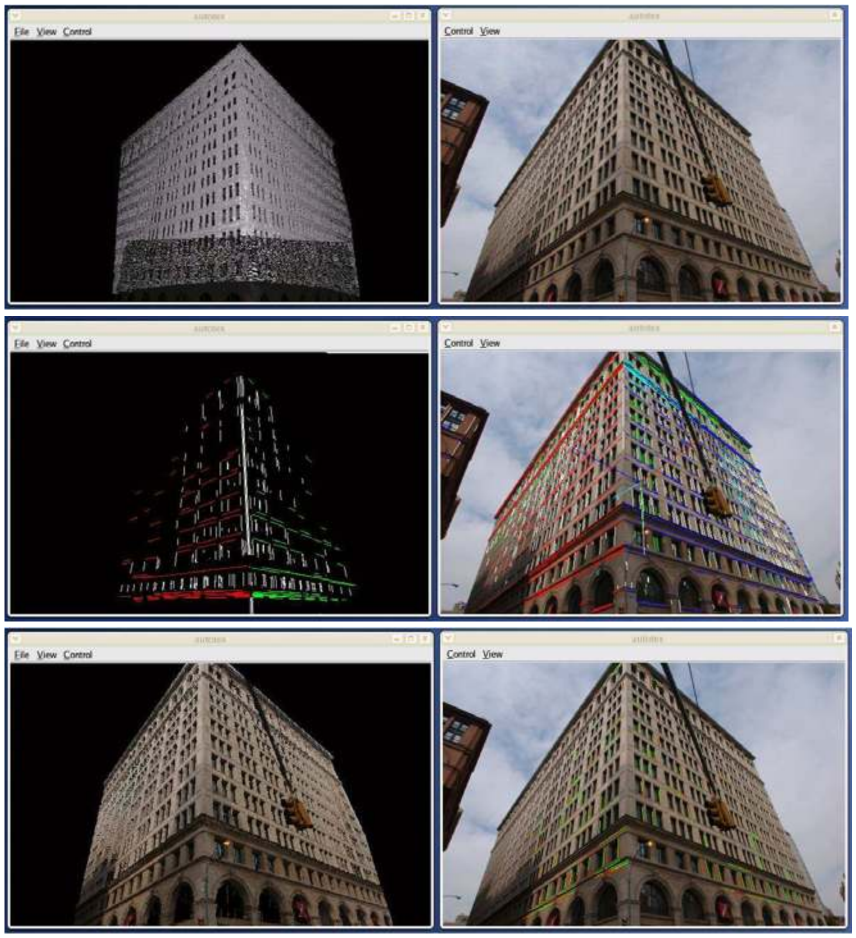
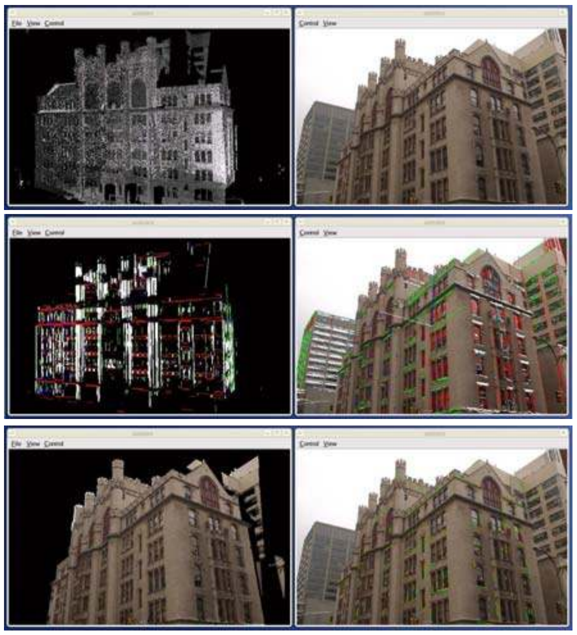


FIG. 14. Registration result of Building 2. **Top row**: Initial state (before registration). The 3D range model (left column) and 2D image (right column) are loaded and displayed in the interface. **Middle row**: The state of the system after the feature extraction. The 3D viewer (left column) shows the clustered 3D lines while the 2D viewer (right column) shows the clustered 2D lines that are drawn on the original 2D image. Different clusters are represented by different colors for clarity. **Bottom row**: The final registration. The 2D image is automatically registered with the 3D range data. The 3D viewer (left) shows the texture mapped 3D range data. The 2D viewer (right) shows the matching 2D and 3D line features (2D lines are displayed as red, while projected 3D lines are highlighted in green). Note that objects that are not part of the 3D model cannot be texture-mapped (corner of other building shown in the 2D image). *A real-time video of the whole process is included as supplemental material.*

step however (Sec. 4.3) depends on the quality of the extracted low-level 2D and 3D linear features. In cases that we cannot extract features of high quality (due to low contrast in 2D images), this method will not be able to perform correctly. On the other hand few correct 2D-to-3D registrations can be enhanced with multiview-geometry solutions to bring sequences in alignment with a model (see Sec. 5).

We have presented a systematic way for registering individual 2D images with a 3D range model. Our methods assume the existence of at least two vanishing points in the scene (not necessarily orthogonal). No higher-order grouping of features is necessary. Our


(a)

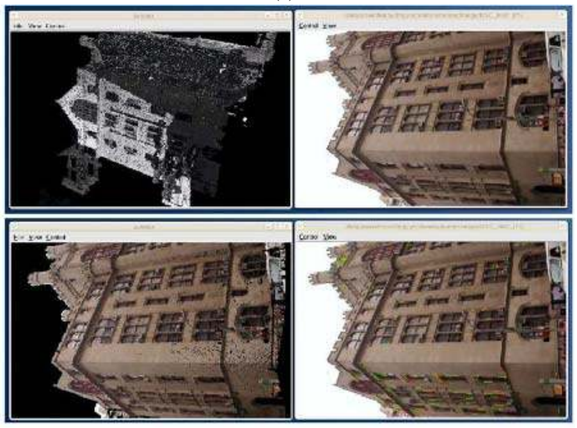
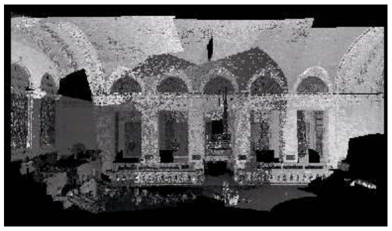
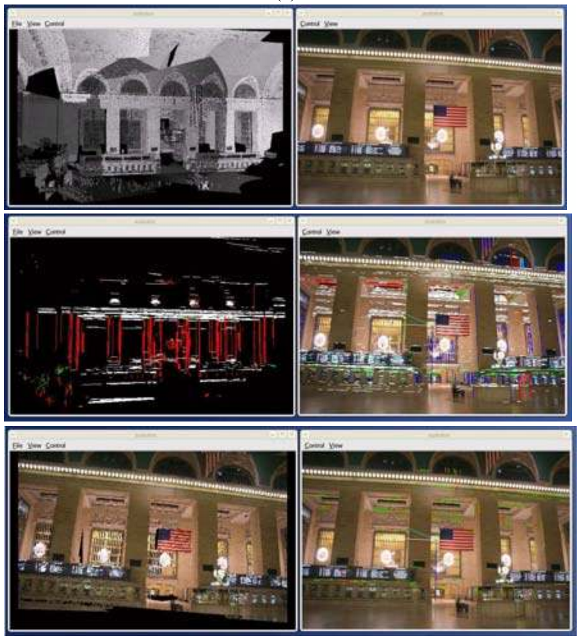

(b)

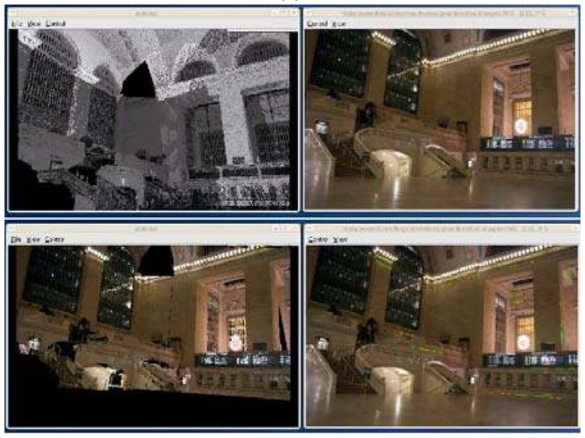
FIG. 15. Registration results from building 1. **(a)** For description see caption of Fig. 14. **(b) (Top row):** The user rotates the 3D model so that it is orientated similarly (note that it does not have to be exactly matched) to the 2D image. **(Bottom row):** The right image shows the 2D image along with the matched 2D and projected 3D features (see caption of Fig. 14). The left image shows the texture-mapped 3D range model after successful registration.



(a)



(b)



(c)

FIG. 16. Registration results from the interior of building 3 (Grand Central Terminal, NYC). **(a)** Registered point cloud of building 3. **(b)** For description see caption of Fig. 14. **(c) (Top row):** The user rotates the 3D model so that it is orientated similarly (note that it does not have to be exactly matched) to the 2D image. **(Bottom row):** The right image shows the 2D image along with the matched 2D and projected 3D features (see caption of Fig. 14). The left image shows the texture-mapped 3D range model after successful registration. Note that surfaces that are not part of the 3D model cannot be texture-mapped and appear as black holes. For example the floor is missing from our range model.

system allow us to register 2D images with a 3D model at interactive rates. In our future work we would like to be able to handle scenes of general configuration not containing any major vanishing points. This would let the exploration of registration algorithms in non-urban scenes.

5. MULTIVIEW POSE ESTIMATION AND 3D STRUCTURE RECONSTRUCTION

The input to our system is a sequence $\mathbf{I} = \{I_n | n = 1, \ldots, N\}$ of high resolution still images that capture the 3D scene. This is necessary to produce photorealistic scene representations. Therefore we have to attack the problem of finding correspondences in a sequence of wide-baseline high resolution images, a problem that is much harder than feature tracking from a video sequence. Fortunately, there are several recent approaches that attack the wide-baseline matching problem [40, 47, 31]. For the purposes of our system, we have adopted the scale-invariant feature transform (SIFT) method [31] for pairwise feature extraction and matching. In general, structure from motion (SFM) from a set images has been rigorously studied [14, 17, 32]. Our method for pose estimation and partial structure recovery is based on sequential updating. The method is similar to work explained in [2, 35]. In order to get very accurate pose estimation, we assume that the camera(s) are pre-calibrated. It is, of course, possible to recover unknown and varying focal length by first recovering pose and structure up to an unknown projective transform and then upgrading to Euclidean space as shown in [19, 34, 45]. However, some of the assumptions that these methods make (e.g., no skew, approximate knowledge of the aspect ratio and principal point) may produce visible mismatches in a high resolution texture map. Thus, for the sake of accuracy we are utilizing the camera calibration method of [51]. More details of our structure-from-motion method can be found in [30].

6. ALIGNMENT OF 2D IMAGE SEQUENCES ONTO 3D-RANGE POINT CLOUDS

The 2D-to-3D registration module described in Sec. 4 facilitates the texturing of the 3D range model M_{range} with photographs taken of the scene. A drawback to this approach is that only a subset of the images are successfully registered with M_{range} and so the texturing is restricted to these images for which the camera pose information is recovered. In order to more fully exploit the use of all images, we opt to use multiview geometry to derive a dense set of camera pose information and a sparse 3D model directly from the images. Merging the results of Sec. 4 with a structure-from-motion algorithm will produce a superior phototextured result that captures the full detail of a complete collection of photographs. We therefore create a sparse 3D model, M_{sfm}, and register it to M_{range}. The rotation, scale, and translation necessary to align these two models also applies to the dense set of camera poses derived from the images. After applying this transformation to the camera poses derived from structure-from-motion, the cameras will sit in the coordinate frame of M_{range} where images can then be projected onto the dense model. The method is described more fully below.

The set of dense range scans $\{S_i | i = 1, \ldots, K\}$ are registered in the same reference frame (Sec. 3), producing a 3D range model called M_{range}. On the other hand, the sequence of 2D images $\mathbf{I} = \{I_n | n = 1, \ldots, N\}$ produces a sparser 3D model of the scene (Sec. 5) called M_{sfm}. Both of these models are represented as clouds of 3D points. The distance between any two points in M_{range} corresponds to the actual distance of the points in

TABLE 3
Building 1 (13 images) - Building 2 (7 images) - Building 3 (6 images). Each row presents results from successful registration of a different 2D image with the 3D range model. The upper part of the table presents results of the registration of 13 images with a 3D range model of building 1. The middle part shows results from registering 7 images with a 3D range model of building 2. Finally, the lower part describes results from the registration of 6 images with the 3D range model of building 3. The registration (matching phase) of each image requires on average 5 to 10 seconds (2GHz Xeon Intel processor, 2GB of RAM). The first two columns show the numbers of 3D and 2D features used for matching. ”Fi” is the initial focal length extracted from the Exif meta-data of the image, while ”Fr” is the refined focal length. ”M” is the number of matched features of the best transformation. Finally, ”E” is the average line-to-line distance (in pixels) after the optimization (Step 6).

F3D	F2D	Fi	Fr	M	E
672	412	3065.83	3072.42	119	4.4492
583	345	3065.83	3075.34	103	4.9394
409	390	3065.83	3071.90	112	4.8973
392	230	3065.83	3069.45	93	4.2109
321	312	3065.83	3073.23	187	4.9021
456	387	3065.83	3072.12	134	4.3902
402	390	3065.83	3071.29	94	3.9827
390	219	3065.83	3069.22	87	4.2023
592	539	3065.83	3071.90	212	4.3003
390	416	3065.83	3061.39	145	3.9203
271	392	3065.83	3073.38	123	3.2900
430	456	3065.83	3076.19	209	4.1293
390	549	3065.83	3063.56	115	4.5902
438	789	1185.03	1165.65	114	4.3215
421	654	1185.03	1175.89	83	4.2142
389	520	1185.03	1172.90	88	3.8992
402	432	1185.03	1179.34	101	4.2390
389	598	1185.03	1172.90	91	4.5009
435	621	1185.03	1169.39	156	4.1290
419	535	1185.03	1178.17	182	4.4923
543	245	2805.81	2833.45	63	4.4439
569	312	2805.81	2831.32	45	3.9082
389	245	2805.81	2829.39	42	4.2312
390	190	2805.81	2839.93	50	4.9821
493	231	2805.81	2812.24	63	3.9023
301	189	2805.81	2829.39	58	3.8910

3D space, whereas the distance of any two points in M_{sfm} is the actual distance multiplied by an unknown scale factor s. In order to align the two models a similarity transformation that includes the scale factor s, a rotation R and a translation T needs to be computed. In this section, a novel algorithm [30] that automatically computes this transformation is presented. The transformation allows for the optimal texture mapping of all images onto the dense M_{range} model, and thus provides photorealistic results of high quality.

Every point X from M_{sfm} can be projected onto a 2D image $I_n \in \mathbf{I}$ by the following transformation:

$$\mathbf{x} = \mathcal{K}_n[R_n \,|\, T_n\,]\,\mathbf{X} \tag{1}$$

where $\mathbf{x} = (x, y, 1)$ is a pixel on image I_n, $\mathbf{X} = (X, Y, Z, 1)$ is a point of M_{sfm}, $\mathcal{K}_n$ is the projection matrix, R_n is the rotation transformation and T_n is the translation vector. These matrices and points $\mathbf{X}$ are computed by the SFM method (Sec. 5).

Some of the 2D images $\mathbf{I}' \subset \mathbf{I}$ are also automatically registered with the 3D range model M_{range} (Sec. 4). Thus, each point of M_{range} can be projected onto each 2D image $I_n \in \mathbf{I}'$ by the following transformation:

$$\mathbf{y} = \mathcal{K}_n[R'_n \,|\, T'_n\,]\,\mathbf{Y} \tag{2}$$

where $\mathbf{y} = (x, y, 1)$ is a pixel in image I_n, $\mathbf{Y} = (X, Y, Z, 1)$ is a point of model M_{range}, $\mathcal{K}_n$ is the projection matrix of I_n, R'_n is the rotation and T'_n is the translation. These transformations are computed by the 2D-to-3D registration method (Sec. 4).

The key idea is to use the images in $I_n \in \mathbf{I}'$ as references in order to find the corresponding points between M_{range} and M_{sfm}. The similarity transformation between M_{range} and M_{sfm} is then computed based on these correspondences. In summary, the algorithm works as follows:

1. Each point of M_{sfm} is projected onto $I_n \in \mathbf{I}'$ using Eq. (1). Each pixel $p_{(i,j)}$ of I_n is associated with the closest projected point $\mathbf{X} \in M_{sfm}$ in an $L \times L$ neighborhood on the image. Each point of M_{range} is also projected onto I_n using Eq. (2). Similarly, each pixel $p_{(i,j)}$ is associated with the projected point $\mathbf{Y} \in M_{range}$ in an $L \times L$ neighborhood (Fig. 17). Z-buffering is used to handle occlusions.
2. If a pixel $p_{(i,j)}$ of image I_n is associated with a pair of 3D points $(\mathbf{X}, \mathbf{Y})$, one from M_{sfm} and the other from M_{range}, then these two 3D points are considered as candidate matches. Thus, for each 2D-image in $\mathbf{I}'$ a set of matches is computed, producing a collection of candidate matches named $\mathbf{L}$. These 3D-3D correspondences between points of M_{range} and points of M_{sfm} could be potentially used for the computation of the similarity transformation between the two models. The set $\mathbf{L}$ contains many outliers, due to the very simple closest-point algorithm utilized. However, $\mathbf{L}$ can be further refined (Sec. 6.1) into a set of robust 3D point correspondences $\mathcal{C} \subset \mathbf{L}$.
3. Finally, the transformation between M_{range} and M_{sfm} is computed by minimizing a weighted error function E (Sec. 6.1) based on the final robust set of correspondences $\mathcal{C}$.

Even though the two models (M_{sfm} and M_{range}) have different noise characteristics, they have similar errors in the operating range that we are using them. We ultimately exploit principles from close-range photogrammetry and therefore we do not use photos that are too far from the object. In that case, points that are recovered that are deemed to be too far away (which also may be due to small baseline) will be dismissed. The remaining projected points from the SFM model can therefore be matched against the projections of the 3D points from the range data.

6.1. Correspondence Refinement and Optimization

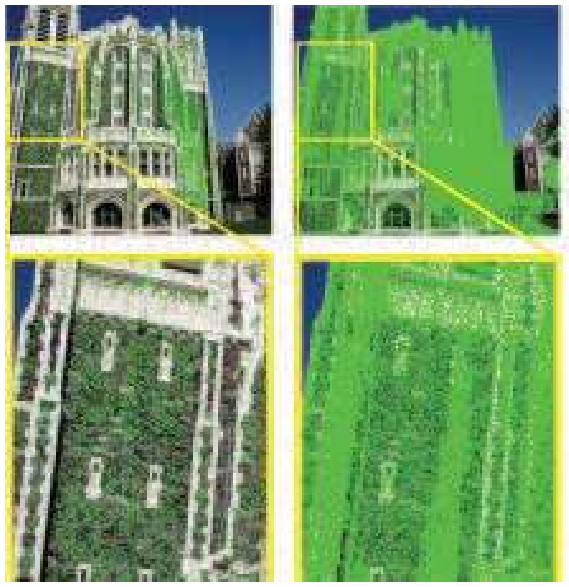


FIG. 17. **Left column**: The points of model M_{sfm} projected onto one 2D image I_n (Sec. 5). The projected points are shown in green. **Right column**: The points of model M_{range} projected onto the same 2D image I_n (projected points shown in green) after the automatic 2D-to-3D registration (Sec. 4). Note that the density of 3D range points is much higher than the density of the SFM points, due to the different nature of the two reconstruction processes. Finding corresponding points between M_{range} and M_{sfm} is possible on the 2D image space of I_n. This yields the transformation between the two models (Sec. 6).

The set of candidate matches $\mathbf{L}$ computed in the second step of the previous algorithm contains outliers due to errors introduced from the various modules of the system (SFM, 2D-to-3D registration, range sensing). It is thus important to filter out as many outliers as possible through verification procedures. A natural verification procedure involves the difference in scale between the two models. Consider two pairs of plausible matched 3D-points $(\mathbf{X_1}, \mathbf{Y_1})$ and $(\mathbf{X_2}, \mathbf{Y_2})$ ($\mathbf{X_i}$ denotes points from the M_{sfm} model, while $\mathbf{Y_j}$ points from the the M_{range} model). If these were indeed correct correspondences, then the scale factor between between the two models would be $s = \|\mathbf{X_1} - \mathbf{X_2}\| / \|\mathbf{Y_1} - \mathbf{Y_2}\|$. Since the computed scale factor should be the same no matter which correct matching pair is used, then a robust set of correspondences from $\mathbf{L}$ should contain only these pairs that produce the same scale factor s. The constant scale factor among correctly picked pairs is thus an invariant feature that we exploit. We now explain how we achieve this robust set of correspondences.

For each image $I_n \in \mathbf{I}'$, let us call the camera's center of projection as $\mathbf{C_n^{sfm}}$ in the local coordinate system of M_{sfm} and $\mathbf{C_n^{rng}}$ in the coordinate system of M_{range}. These two centers have been computed from two independent processes: SFM (Sec. 5) and 2D-to-3D registration (Sec. 4). Then for any candidate match, $(\mathbf{X}, \mathbf{Y}) \in \mathbf{L}$, a candidate scale factor

$s_1(\mathbf{X}, \mathbf{Y})$ can be computed as:

$$s_1(\mathbf{X}, \mathbf{Y}) = \frac{\|\mathbf{X} - \mathbf{C_n^{sfm}}\|}{\|\mathbf{Y} - \mathbf{C_n^{rng}}\|}$$

If we keep the match $(\mathbf{X}, \mathbf{Y})$ fixed and we consider every other match $(\mathbf{X}', \mathbf{Y}') \in \mathbf{L}$, $L-1$ candidate scale factors $s_2(\mathbf{X}', \mathbf{Y}')$ and $L-1$ candidate scale factors $s_3(\mathbf{X}', \mathbf{Y}')$ (L is the number of matches in $\mathbf{L}$) are computed as:

$$s_2(\mathbf{X}', \mathbf{Y}') = \frac{\|\mathbf{X}' - \mathbf{C_n^{sfm}}\|}{\|\mathbf{Y}' - \mathbf{C_n^{rng}}\|}, s_3(\mathbf{X}', \mathbf{Y}') = \frac{\|\mathbf{X} - \mathbf{X}'\|}{\|\mathbf{Y} - \mathbf{Y}'\|}$$

That means that if we keep the match $(\mathbf{X}, \mathbf{Y})$ fixed, and consider all other matches $(\mathbf{X}', \mathbf{Y}')$ we can compute a triple of candidate scale factors: $s_1(\mathbf{X}, \mathbf{Y})$, $s_2(\mathbf{X}', \mathbf{Y}')$, and $s_3(\mathbf{X}', \mathbf{Y}')$. We then consider the two pairs of matches $(\mathbf{X}, \mathbf{Y})$ and $(\mathbf{X}', \mathbf{Y}')$ as *compatible* if the scale factors in the above triple are close with respect to each other. By fixing $(\mathbf{X}, \mathbf{Y})$, all matches that are compatible with it are found. The confidence in the match $(\mathbf{X}, \mathbf{Y})$ is the number of compatible matches it has. By going through all matches in $\mathbf{L}$, their confidence is computed via the above procedure. Out of these matches the one with the highest confidence is selected as the most prominent: $(\mathbf{X_p}, \mathbf{Y_p})$. Let us call $\mathbf{L_n}$ the set that contains $(\mathbf{X_p}, \mathbf{Y_p})$ and all other matches that are compatible with it. Note that this set is based on the centers of projection of image I_n as computed by SFM and 2D-to-3D registration. Let us also call s_n the scale factor that corresponds to the set $\mathbf{L_n}$. This scale factor can be computed by averaging the triples of scale factors of the elements in $\mathbf{L_n}$. Finally a different set $\mathbf{L_n}$ and scale factor s_n is computed for every image $I_n \in \mathbf{I}'$.

From the previous discussion it is clear that each $\mathbf{L_n}$ is a set of matches that is based on the center of projection of each image I_n independently. A set of matches that will provide a globally optimal solution should consider all images of $\mathbf{I}'$ simultaneously. Out of the scale factors computed from each set $\mathbf{L_n}$, the one that corresponds to the largest number of matches is the one more robustly extracted by the above procedure. That computed scale factor, s_{opt}, is used as the final filtration for the production of the robust set of matches $\mathcal{C}$ out of $\mathbf{L}$. In particular, for each candidate match $(\mathbf{X}, \mathbf{Y}) \in \mathbf{L}$, a set of scale factors are computed as

$$s'_n = \frac{\|\mathbf{X} - \mathbf{C_n^{sfm}}\|}{\|\mathbf{Y} - \mathbf{C_n^{rng}}\|}$$

where $n = 1, 2, ..., J$, and J is the number of images in $\mathbf{I}'$. The standard deviation of those scale factors with respect to s_{opt} is computed and if it is smaller than a user-defined threshold, $(\mathbf{X}, \mathbf{Y})$ is considered as a *robust* match and is added to the final list of correspondences $\mathcal{C}$. The robustness of the match stems from the fact that it verifies the robustly extracted scale factor s_{opt} with respect to most (or all) images $I_n \in \mathbf{I}'$. The pairs of center of projections $(\mathbf{C_n^{sfm}}, \mathbf{C_n^{rng}})$ of images in $\mathbf{I}'$ are also added to $\mathcal{C}$.

The list $\mathcal{C}$ contains robust 3D point correspondences that are used for the accurate computation of the similarity transformation (scale factor s, rotation R, and translation T) between the models M_{range} and M_{sfm}. The following weighted error function is minimized with respect to sR and T:

$$E = \sum_{(\mathbf{X}, \mathbf{Y}) \in \mathcal{C}} w \|sR \cdot \mathbf{Y} + T - \mathbf{X}\|^2$$

where the weight $w = 1$ for all $(\mathbf{X}, \mathbf{Y}) \in \mathcal{C}$ that are not the centers of projection of the cameras, and $w > 1$ (user-defined) when $(\mathbf{X}, \mathbf{Y}) = (\mathbf{C_n^{sfm}}, \mathbf{C_n^{rng}})$. By associating higher weights to the centers we exploit the fact that we are confident in the original pose produced by SFM and 2D-to-3D registration. The unknown sR and T are estimated by computing the least square solution from this error function. Note that s can be easily extracted from sR since the determinant of R is 1.

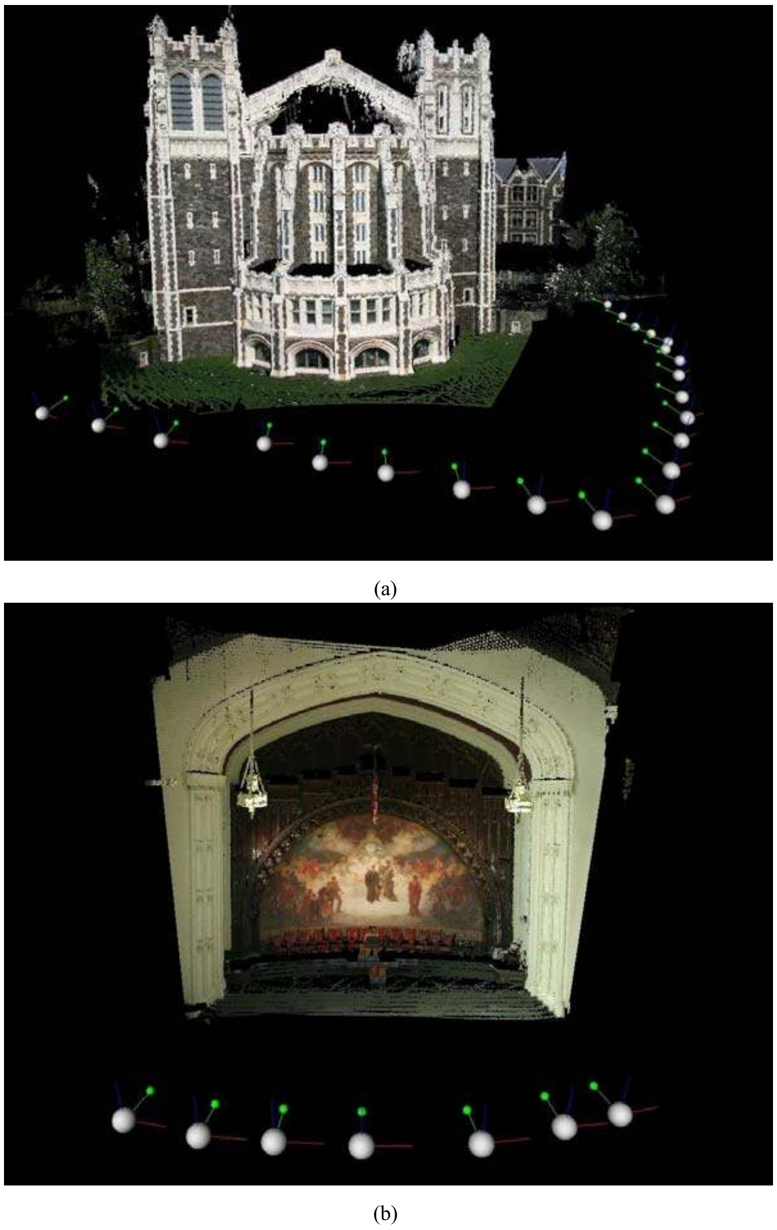
In summary, by utilizing the invariance of the scale factor between corresponding points in M_{range} and M_{sfm}, a set of robust 3D point correspondences $\mathcal{C}$ is computed. These 3D point correspondences are then used for an optimal calculation of the similarity transformation between the two point clouds. This provides a very accurate texture mapping result of the high resolution images onto the dense range model M_{range}.

7. RESULTS & CONCLUSIONS

We tested our algorithms using range scans and 2D images acquired from a large-scale urban structure (Shepard Hall/CCNY) and from an interior scene (Great Hall/CCNY). 22 range scans of the exterior of Shepard Hall were automatically registered (Fig. 1) to produce a dense model M_{range}. In one experiment, ten images where gathered under the same lighting conditions. All ten of them were independently registered (2D-to-3D registration Sec. 4) with the model M_{range}. The registration was optimized with the incorporation of the SFM model (Sec. 5) and the final optimization method (Sec. 6). In a second experiment, 22 images of Shepard Hall that covered a wider area were acquired. Although the automated 2D-to-3D registration method was applied to all the images, only five of them were manually selected for the final transformation (Sec. 6) on the basis of visual accuracy. For some of the 22 images the automated 2D-to-3D method could not be applied due to lack of linear features. However, all 22 images where optimally registered using our novel registration method (Sec. 6) after the SFM computation (Sec. 5). Fig. 1 shows the alignment of the range and SFM models achieved through the use of the 2D images. In Fig. 18(a) the accuracy of the texture mapping method is visible. Fig. 18(b) displays a similar result of an interior 3D scene [9]. Table 4 provides some quantitative results of our experiments. Notice the density of the range models versus the sparsity of the SFM models. Also notice the number of robust matches in $\mathcal{C}$ (Sec. 6) with respect to the possible number of matches (i.e., number of points in SFM). The final row Table 4 displays the elapsed time for the final optimization on a Dell PC running Linux on an Intel Xeon-2GHz, 2GB-RAM machine.

We have presented a system that integrates multiview geometry and automated 3D registration techniques for texture mapping high resolution 2D images onto dense 3D range data. The benefits of multiview geometry (SFM) and automated 2D-to-3D/3D-to-3D registration are merged for the production of photorealistic models with minimal human interaction. Our approach provides a solution of increased robustness, efficiency and generality with respect to previous methods.

[9] `http://www.cs.hunter.cuny.edu/~ioannis/IjcvSI` contains images and videos of our results.



(a)

(b)

FIG. 18. **(a)** Range model of Shepard Hall (CCNY) with 22 automatically texture mapped high resolution images. **(b)** Range model of interior scene (Great Hall) with seven automatically texture mapped images. The locations of the recovered camera positions are shown. Notice the accuracy of the photorealistic result.

TABLE 4
Quantitative results.

	Shepard Hall		Great Hall
Number of points (M_{range})	12,483,568		13,234,532
Number of points (M_{sfm})	2,034	45,392	1,655
2D-images used	10	22	7
2D-to-3D registrations (Sec. 4)	10	5	3
No. of matches in $\mathcal{C}$ (Sec. 6)	258	1632	156
Final optimization (Sec. 6)	8.65 s	19.20 s	3.18 s

ACKNOWLEDGMENTS

This work was supported in part by the following grants: NSF CAREER IIS-02-37878, NSF MRI/RUI EIA-0215962, and DOE DE-FG52-06NA27503.

REFERENCES

1. A. P. Ambler, H. G. Barrow, C. M. Brown, R. M. Burstall, and R. J. Popplestone. A versatile computer-controlled assembly system. *The 3rd International Joint Conference on Artificial Intelligence*, 1973.
2. P. A. Beardsley, A. P. Zisserman, and D. W. Murray. Sequential updating of projective and affine structure from motion. *International Journal of Computer Vision*, 23(3):235–259, 1997.
3. O. R. P. Bellon and L. Silva. New improvements to range image segmentation by edge detection. *IEEE Signal Processing Letters*, 9(2):43–45, February 2002.
4. F. Bernardini and H. Rushmeier. The 3D model acquisition pipeline. *Computer Graphics Forum*, 21(2):149–172, June 2002.
5. P. J. Besl and R. C. Jain. Segmentation through variable-order surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(2):167–192, March 1988.
6. P. J. Besl and N. D. McKay. A method for registration of 3D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2), 1992.
7. T. Cass. Polynomial–time geometric matching for object recognition. *International Journal of Computer Vision*, 21(1–2):37–61, 1997.
8. C. Chen. *Range Segmentation and Registration for 3D Modeling of Large-Scale Urban Scenes*. PhD thesis, City University of New York, 2007.
9. C. Chen and I. Stamos. Semi-automatic range to range registration: A feature-based method. In *The 5th International Conference on 3-D Digital Imaging and Modeling*, pages 254–261, Ottawa, June 2005.
10. C. Chen and I. Stamos. Range image registration based on circular features. In *Intl. Symposium on 3D Data Processing, Visualization and Transmission*, Chapel Hill, June 2006.
11. S. Christy and R. Horaud. Iterative pose computation from line correspondences. *Journal of Computer Vision and Image Understanding*, 73(1):137–144, January 1999.
12. G. L. D. Marshall and R. Martin. Robust segmentation of primitives from range data in the presence of geometric degeneracy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):304–314, March 2001.
13. O. Faugeras. *Three–Dimensional Computer Vision*. The MIT Press, 1996.
14. O. Faugeras, Q. T. Luong, and T. Papadopoulos. *The Geometry of Multiple Images.* MIT Press, 2001.
15. M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Graphics and Image Processing*, 24(6):381–395, June 1981.
16. C. Früh and A. Zakhor. Constructing 3D city models by merging aerial and ground views. *Computer Graphics and Applications*, 23(6):52–11, 2003.

17. R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision, second edition*. Cambridge University Press, 2003.

18. G. Hausler and D. Ritter. Feature–based object recognition and localization in 3D-space, using a single video image. *Journal of Computer Vision and Image Understanding*, 73(1):64–81, 1999.

19. A. Heyden and K. Astrom. Euclidean reconstruction from constant intrinsic parameters. In *International Conference on Pattern Recognition*, volume 1, pages 339–343, 1992.

20. R. Horaud, F. Dornaika, B. Lamiroy, and S. Christy. Object pose: The link between weak perspective, paraperspective, and full perspective. *International Journal of Computer Vision*, 22(2), 1997.

21. D. F. Huber and M. Hebert. Fully automatic registration of multiple 3D data sets. *Image and Vision Computing*, 21(7):637–650, July 2003.

22. D. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, 5(7):195–212, 1990.

23. K. Ikeuchi. The great buddha project. In *IEEE ISMAR03*, Tokyo, Japan, November 2003.

24. D. W. Jacobs. Matching 3–D models to 2–D images. *International Journal of Computer Vision*, 21(1–2):123–153, 1997.

25. A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, 1999.

26. F. Jurie. Solution of the simultaneous pose and correspondence problem using gaussian error model. *Journal of Computer Vision and Image Understanding*, 73(3):357–373, March 1999.

27. Leica Geosystems. http://hds.leica-geosystems.com/.

28. L. Liu. *Automated Registration of 2D Images with 3D Range Data in a Photorealistic Modeling System of Urban Scenes*. PhD thesis, City University of New York, 2007.

29. L. Liu and I. Stamos. Automatic 3D to 2D registration for the photorealistic rendering of urban scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume II, pages 137–143, San Diego, CA, June 2005.

30. L. Liu, I. Stamos, G. Yu, G. Wolberg, and S. Zokai. Multiview geometry for texture mapping 2D images onto 3D range data. In *IEEE Conf. Computer Vision and Pattern Recognition*, volume II, pages 2293–2300, New York City, June 2006.

31. D. G. Lowe. Distinctive image features from scale-invariant keypoint. *International Journal of Computer Vision*, 60(2):91–110, 2004.

32. Y. Ma, S. Soatto, J. Kosecka, and S. Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models.* Springer-Verlag, 2003.

33. D. Oberkampf, D. DeMenthon, and L. Davis. Iterative pose estimation using coplanar feature points. *Computer Vision Graphics and Image Processing*, 63(3), May 1996.

34. M. Pollefeys and L. V. Gool. A stratifed approach to metric self-calibration. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 407–412, 1997.

35. M. Pollefeys, L. V. Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera. *International Journal of Computer Vision*, 59(3):207–232, 2004.

36. K. Pulli, H. Abi-Rached, T. Duchamp, L. G. Shapiro, and W. Stuetzle. Acquisition and visualization of colored 3D objects. In *International Conference on Pattern Recognition*, volume 1, page 11, Australia, 1998.

37. K. Pulli and M. Pietikinen. Range image segmentation based on decomposition of surface normals. In *Proceedings of the Scandinavian Conference on Image Analysis*, 1993.

38. L. Quan and Z. Lan. Linear N–point camera pose determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(7), July 1999.

39. S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *The 3rd International Conference on 3-D Digital Imaging and Modeling*, June 2001.

40. F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo. In *IEEE International Conference on Computer Vision*, pages 636–643, July 2001.

41. V. Sequeira and J. Concalves. 3D reality modeling: Photo-realistic 3D models of real world scenes. In *Intl. Symposium on 3D Data Processing, Visualization and Transmission*, pages 776–783, 2002.

42. I. Stamos and P. K. Allen. Automatic registration of 3-D with 2-D imagery in urban environments. *IEEE International Conference on Computer Vision*, pages 731–736, 2001.

43. I. Stamos and P. K. Allen. Geometry and texture recovery of scenes of large scale. *Journal of Computer Vision and Image Understanding*, 88(2):94–118, 2002.

44. I. Stamos and M. Leordeanu. Automated feature-based range registration of urban scenes of large scale. *IEEE Conference on Computer Vision and Pattern Recognition*, 2:555–561, 2003.

45. B. Triggs. Factorization methods for projective structure and motion. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 845–851, 1996.

46. A. Troccoli and P. K. Allen. A shadow based method for image to model registration. In *2nd IEEE Workshop on Video and Image Registration*, July 2004.

47. T. Tuytelaars and L. J. V. Gool. Matching widely separated views based on affine invariant regions. *International Journal of Computer Vision*, 59(1):61–85, 2004.

48. Visual Information Technology Group, Canada, 2005. http://iit-iti.nrc-cnrc.gc.ca/about-sujet/vit-tiv_e.html.

49. M. A. Wami and B. G. Batchelor. Edge-region-based segmentation of range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(3):314–319, March 1994.

50. W. Wells. Statistical approaches to feature–based object recognition. *International Journal of Computer Vision*, 21(1–2):63–98, 1997.

51. Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

52. H. Zhao and R. Shibasaki. Reconstructing a textured CAD model of an urban environment using vehicle-borne laser range scanners and line cameras. *Machine Vision and Applications*, 14(1):35–41, 2003.

53. W. Zhao, D. Nister, and S. Hsu. Alignment of continuous video onto 3D point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1305–1318, 2005.