
AVNTK SC


# 3D Reconstruction from FMV for terrain awareness

Fidel Gutiérrez-Resendiz

AVNTK S.C.
Av. Chapalita 1143, Col. Chapalita, C.P. 45040 Guadalajara, Jalisco, México.
fidel@avntk.com

**Abstract**—*We have developed an advanced system for 3D scene reconstruction for terrain awareness from multiple image views, such as those obtain from full-motion videos (FMV) onboard unmanned vehicles (UAVs). If a scene of interest is seen from several viewing cameras (with different calibration parameters and physical characteristics) positioned around the target, even distinct illumination degrees, occlusion, scale, image resolution, share effect or motion, we have the only commercially available computational package capable of getting a high-definition 3D scene reconstruction, as part of our Raphael Ground Control Station (GCS) system. The system is able to reconstruct 3D targets with an accuracy of <1% relative to a characteristic length such as building height, where said targets are reconstructed from a series of FMV sequences. Our system overcomes multiple texture problem, given by distinct illumination conditions, imaging systems and mission tracks. However, we exploit multiple mission tracks in order to get a more accurate 3D scene reconstruction.*

**Index Terms**—*3D Reconstruction, Raphael Ground Control Station, UAV.*

## 1. Introduction

This 3D scene reconstruction work for multiple image views of a target of interest, where the position of each viewing image is only roughly known, is ideal when digital images (including photographs or video) are available, for example because the phenomenon is unique or of difficult access such as an asteroid, perhaps videos from youtube website, or a combination of aerial videos taken at different times from varying positions with a range of imaging systems and/or ground images of a target of interest. When a sophisticated, expensive and heavy sensor such as a Light Detection And Ranging technology (LIDAR) or SAR cannot be readily accessed due to cost, availability or maximum payload constraints of a platform such as a tactical UAV, even thus, we can get an accurate 3D scene reconstruction using video/images. This approach will be widely useful in coming years.

Initially, our 3D scene reconstruction algorithm assumes multiple image views in the form of several digitally-captured 2D images of a scene or target taken from different angles and under distinct weather conditions. Theses images can be coded in .bmp, .png, .jpg, .gif, .mpeg, .avi, or any other digital format. Our software automatically computes information about location and orientation of the cameras only from the images themselves using computer vision methods, though providing rough GPS information certainly speeds up the process.

The algorithm developed works by breaking down the problem into several stages. These will be covered as follows: in *Section 2* camera model and calibration is presented. Detected features points are described on *Section 3*, and *Section 4* presents the algorithm about matching point features between pairs of images. *Section 5* refers to an iterative Structure from Motion (SfM) procedure to recover the camera parameters. *Section 6* presents two methods about stereo-matching of calibrated images. The Poisson reconstruction with oriented 3D reconstructed points set is presented in *Section 7*. The final texture 3D scene is described in *Section 8*. Finally, *Section 9* shows an optional and complementary module that is useful to align multiple mission tracks under distinct weather condition and illumination conditions in order to get a more detailed 3D scene reconstruction model. Figure 1 shows images taken by the synthetic vision module of our *Raphael* Ground Control Station (***Raphael GCS***) (Gutiérrez-Resendiz and Funes-Gallanzi, 2009).

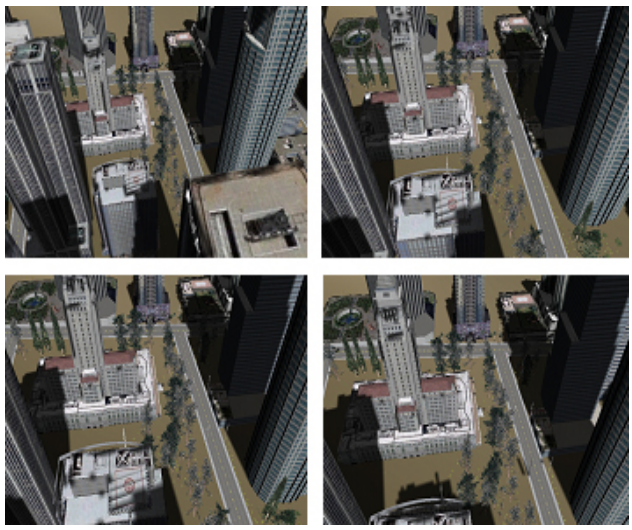


Figure 1.- Four images taken from the synthetic vision toolkit of Raphael GCS are shown.

## 2. Camera Model and Calibration

As a first step, we consider a camera as an electronic device that records an image digitally. The pinhole camera model is widely used as the real camera projection. This model is based on the

principle of collinearity, where each point in the object space is projected by a straight line through the projection centre into the image plane. It is a useful model that enables simple mathematical formulation for the relationship between object and image coordinates. However, it is not valid when high accuracy is required and therefore, a more comprehensive camera model must be used. The model contains parameters divided into extrinsic and intrinsic parameters. Extrinsic parameters refer to camera pose and intrinsic parameters refer to image plane properties.

The origin of the camera coordinate system is in the projection centre at the location $(x_0, y_0, z_0)$ with respect to the object coordinate system, and the z-axis of the camera frame is perpendicular to the image plane. The rotation is represented using Euler angles $\omega, \wp$ and $\kappa$ that define a sequence of three elementary rotations around *x*, *y*, *z*-axis respectively. Rotations are performed clockwise, first around the *x*-axis, then the *y*-axis which is already once rotated, and finally around the *z*-axis that is twice rotated during the previous stages.

In order to express an arbitrary object point *P* at location $(X_i, Y_i, Z_i)$ in image coordinates, we first need to transform it to camera coordinates $(x_i, y_i, z_i)$. This transformation consists of a translation and a rotation, and it can be performed by using the following matrix equation:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \quad (1)$$

where

$$r_{12} = \sin\omega \sin\wp \cos\kappa - \cos\omega \sin\kappa \qquad r_{11} = \cos\wp$$
$$r_{22} = \sin\omega \sin\wp \sin\kappa + \cos\omega \cos\kappa \qquad r_{21} = \cos\wp$$
$$r_{13} = \cos\omega \sin\wp \cos\kappa + \sin\omega \sin\kappa \qquad r_{31} = -s$$
$$r_{23} = \cos\omega \sin\wp \sin\kappa - \sin\omega \cos\kappa \qquad r_{32} = \sin\omega$$

(2)

The intrinsic camera parameters usually include the effective focal length *f*, scale factor *su*, and the image center $(u_0, v_0)$, also called the principal point. Here, as usual in computer vision literature, the origin of the image coordinate system is in the upper left corner of the image array. The unit of the image coordinates is pixel, and therefore coefficients *Du* and *Dv* are needed to change the metric units to pixels. In fact, their precise values are not necessary, because they are linearly dependent on the focal length *f* and the scale factor *su*. By using the pinhole model, the projection of the point $(x_i, y_i, z_i)$ to the image plane is expressed as:

$$\begin{bmatrix} \tilde{u}_i \\ \tilde{v}_i \end{bmatrix} = \frac{f}{z_i} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad (3)$$

The corresponding image coordinates $(u_i', v_i')$ in pixels are obtained from the projection $(\tilde{u}_i, \tilde{v}_i)$ by applying the following transformation:

$$\begin{bmatrix} u_i' \\ v_i' \end{bmatrix} = \begin{bmatrix} D_u s_u \tilde{u}_i \\ D_v \tilde{v}_i \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \quad (4)$$

Usually, the pinhole model is a basis that is extended with some corrections for the systematically distorted image coordinates. The most commonly used correction is for the radial lens distortion that causes the actual image point to be displaced radially in the image plane. The radial distortion can be approximated using the following expression:

$$\begin{bmatrix} \Delta u_i^{(radial)} \\ \Delta v_i^{(radial)} \end{bmatrix} = \begin{bmatrix} \tilde{u}_i \left(k_1 r_i^2 + k_2 r_i^4\right) \\ \tilde{v}_i \left(k_1 r_i^2 + k_2 r_i^4\right) \end{bmatrix} \quad (5)$$

where *k1* and *k2* are coefficients for radial distortion, and $r_i \equiv \sqrt{\tilde{u}_i^2 + \tilde{v}_i^2}$. Typically, one or two coefficients are enough to compensate for the radial distortion.

Curvature centres of surface lenses of lens surface are not always strictly collinear. This introduces another common distortion type, de-centering distortion that has both a radial and tangential component.

The expression for the tangential distortion is often written in the following form:

$$\begin{bmatrix} \Delta u_i^{(tangential)} \\ \Delta v_i^{(tangential)} \end{bmatrix} = \begin{bmatrix} 2p_1 \tilde{u}_i \tilde{v}_i + p_2\left(r_i^2 + 2\tilde{u}_i^2\right) \\ p_1\left(r_i^2 + 2\tilde{v}_i^2\right) + 2p_2 \tilde{u}_i \tilde{v}_i \end{bmatrix} \quad (6)$$

where $p_1$ and $p_2$ are coefficients for tangential distortion.

A sufficiently accurate camera model for accurate calibration can be derived by combining the pinhole model with the correction for the radial and tangential distortion components:

$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} D_u s_u\left(\tilde{u}_i + \Delta u_i^{(radial)} + \Delta u_i^{(tangential)}\right) \\ D_v\left(\tilde{u}_i + \Delta v_i^{(radial)} + \Delta v_i^{(tangential)}\right) \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix}$$

(7)

In this model the set of intrinsic parameters $(f, s_u, u_o, v_0)$ is augmented with the distortion coefficients $k_1$, $k_2$, $p_1$ and $p_2$. These parameters are also known as physical camera parameters, since they have a certain physical meaning. Generally, the objective of the explicit camera calibration procedure is to determine optimal values for these parameters based on image observations of a known 3D target. In the case of self-calibration, the 3D coordinates of the target points are also included in the set of unknown parameters.

Since we have the pinhole camera model for the captured images and we have calibrated each camera, we created an image list with information about image sizes and camera focal length. Since images had been captured under different conditions; this last data could not always exist, however our system may calculate an initial estimation for it, or posses a database with a family of imaging systems characterized as used in any particular theater of operations.

#### 2.1- Numerical example for camera calibration module

In order to calibrate any camera, we have used a chessboard pattern of known dimensions. Each black/white square on the chessboard has 3cm x 3cm of size. Thus, we have used 10 images with 640 x 512 pixel size. These images were taken from distinct angular directions. The calibration algorithm given by Heikkila (1997) gives the following results:

Principal point: 321.9588, 237.3126 pixel.
Scale factor: 0.9989
Effective focal length: 3.2864 mm

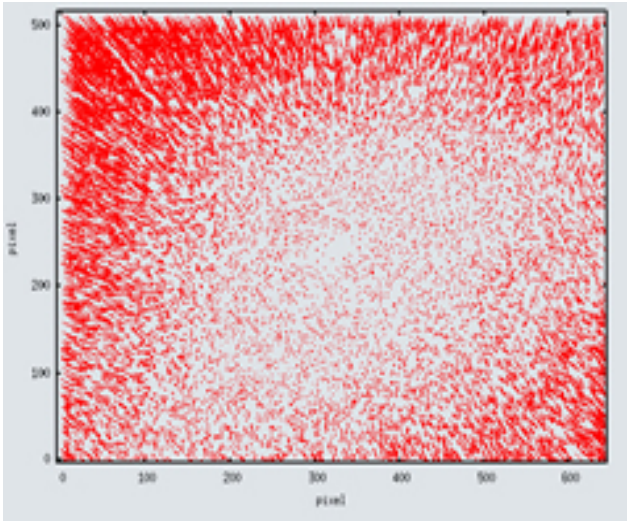



Figure 2. The radial and tangential effects are shown. The effect has been exaggerated 10 times.

Radial distortion coefficient:

$$k_1 = 2.422831 \times 10^{-3}\ mm^{-2}$$

$$k_2 = -1.234360 \times 10^{-3}\ mm^{-4}$$

tangential distortion coefficient:

$$p_1 = 1.343740 \times 10^{-3}\ mm^{-1}$$

$$p_2 = -7.714951 \times 10^{-4}\ mm^{-1}$$

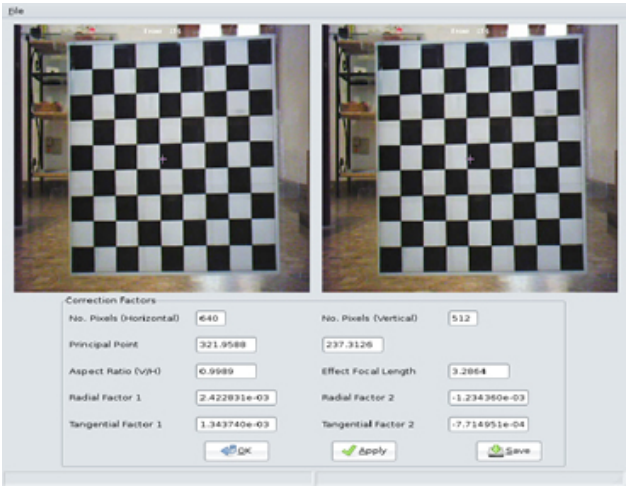



Figure 3. A window of the camera calibration module is showed.

The figures 2 shows camera distortion effect of a typical image system. The effect has been exaggerate 10 times for visual propose only. ***Raphael GCS*** takes into account camera defects very importantly. Therefore, it includes a camera calibration module which fundamental step in 3D

scene reconstruction scheme. A window of ***Raphael GCS*** camera calibration module is showed in figure 3.

### 3. Detection Features

In order to find feature points in each image, we have used feature-matching algorithm, which is a fundamental step in 3D scene reconstruction. We used the well-known SIFT method which is *Scale-Invariant Feature Transform* to detect and describe local features (keypoints) in images (Lowe, 2004). A typical image contains several thousand SIFT keypoints. However, sometimes the number of keypoints falls below the minimum number of keypoints to get a good estimation of initial camera pose. Then, we have adapted an interface editor to increase or modify the number of keypoints, the user can do this by hand. Each keypoint is invariant to image scale, rotation, affine distortion, change in 3D viewpoint, in addition of noise and changes in illumination. The SIFT method ensures that a single feature can be correctly matched with high probability against a large database of feature from many images. The following are the major stages of computation used to generate the set of keypoints:

1. **Scale-space extreme detection**: The first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function to identify interest points that are invariant to scale and orientation.

2. **Keypoint location**: At each candidate location, a detailed model is fitted to determine location and scale. Keypoints are selected based on measures of their stability.

3. **Orientation assignment**: One or more orientations are assigned to each keypoint location based on local image gradient directions. All future operations are performed on image data that have been transformed relatively to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.

4. **Keypoint descriptor**: The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination.

In addition to the keypoints location themselves, SIFT provides a local descriptor for each keypoint in the following sense:

- Consider a 16x16 patch centred on the keypoint.
- Decompose this patch into 4x4 pixel tiles.
- For each such tile, we compute a histogram of its pixels gradient orientations with 8 bins, each covering 45 degrees.
- Actually, we specify these gradient orientations “relative” to the keypoint's dominant orientation.
- We weigh the contribution to the bin by the gradient magnitude.
- We use a circular Gaussian falloff from the keypoint centre (about 8 pixels, half the descriptor window).
- This gives 16 tiles x 8 histogram bins/tile = a 128-dimensional vector representing our a SIFT feature descriptor.
- Normalise this to unit length.
- For better illumination invariance a threshold value in this normalised vector to be no larger than 0.2 and then renormalise.
- This vector normalised is invariant to location, scale, rotation, share and changes in illumination.

The figure 4 shows a typical output of a SIFT detection process. Each image contains thousands of SIFT features. Keypoints are shown as vectors in cyan with magnitude and direction only.

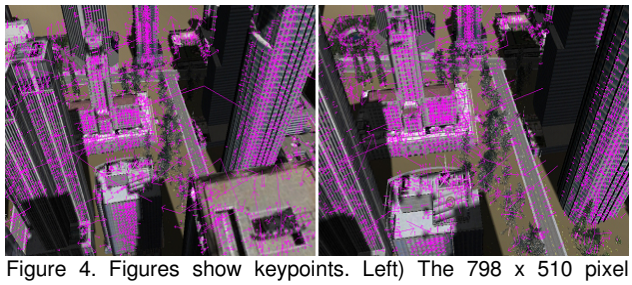


Figure 4. Figures show keypoints. Left) The 798 x 510 pixel image has 4467 keypoints. Right) The 798 x 510 pixel image has 3824 keypoints. The keypoints are displayed as vectors indicating scale, orientation and locations.

### 4. SIFT matching between pair images

Since each keypoint is described by a vector 128-dimensional. We have used the SIFT method to get an automatic feature matching over all image pairs, using the “Approximate Nearest Neighbours” package of Arya, et. al. (1998):

- We take Euclidean distance in the space of 128-dimensional vectors.
- We use kd-trees for efficient near neighbour search.
- We compare the distance to best matching SIFT feature in the database, and the distance to the 2nd-best matching feature. If the ratio of closest distance to 2nd closest distance greater than 0.8 then reject as a false match.

The figure 5 shows a typical example of SIFT matching between two arbitrary chosen images. A cyan line marks the correspondence between two keypoints. However, we have included epipolar geometry constraints in order to do a best matching between keypoints as describe below.

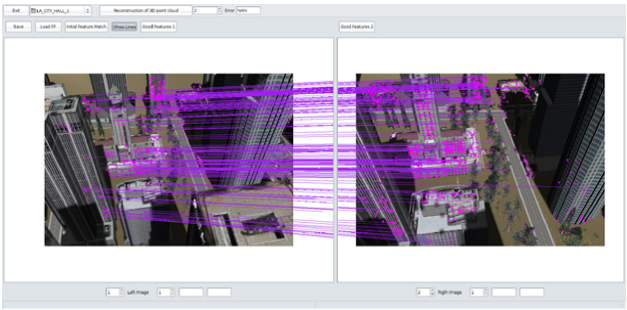


Figure 5. SIFT matching between image 1 and image 2. The cyan line marks the same point seen on different frames.

Once initial matching is done using a local description vector for each keypoint, then robustly we estimate the fundamental matrix *F* for each pair image using the RANSAC method (Fischler and Bolles, 1987). As is well-known in computer vision, the fundamental matrix *F* is a 3 x 3 matrix which relates 2D corresponding points in stereo images. The fundamental matrix relates 2D coordinate images between two stereo images: $x_2 F x_1 = 0$ .

Furthermore, if we know image camera parameters given by $K_1$ and $K_2$ matrices, we have the so-called essential matrix $E = K_2^T F K_1$ , that relates to rotation and translation between cameras: $E = Rt_{[x]}$ .

During each RANSAC iteration, we compute a candidate fundamental matrix *F* using the eight-point algorithm (Hartley and Zisserman 2004), followed by a non-linear refinement. The epipolar constraint is applied intensively around the SIFT matching to ensure better feature matches. We look for a set of geometrically consistent matches between each image pair in an incremental scheme to reconstruct an initial sparse 3D point-cloud and camera pose. Finally, we remove matches that are outliers to the epi-polar geometry constraint. In addition, we recover an initial set of camera parameters and 3D scene reconstructed point-cloud. In order to get refinement for new camera poses and 3D scene point-cloud, we take the back-projection error as the cost function to minimise it. This is a much more computationally-intensive approach than previously reported but it is robust and able to cope with real-life images.

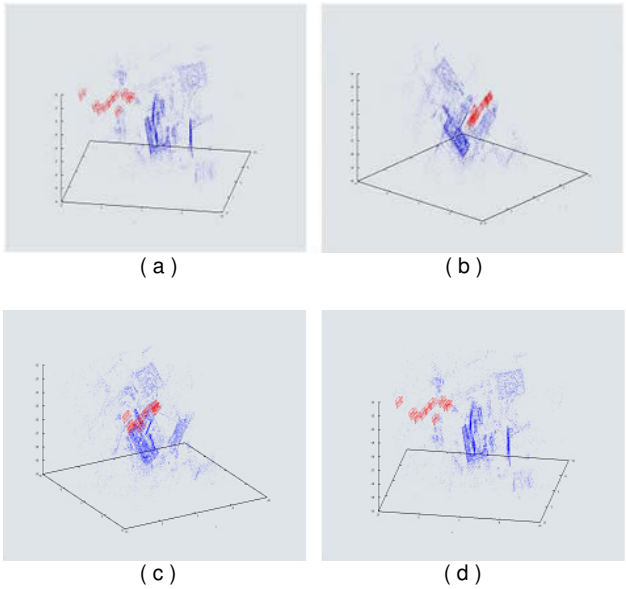



Figure 6. Shows output from a typical Bundle Adjustment run. Figures a), b), c) and d) show 3D point-cloud reconstruction in blue dots and camera pose in red line. We have included 10 camera poses only, for illustration purposes See the full video on http://www.avntk.com

### 5. Bundle adjustment

Once the last frame is added, a regularisation method for final camera poses is calculated by a generic **Bundle Adjustment** method, based on the sparse Levenberg-Marquardt algorithm given by Lourakis and Argyros (2004). The Bundle Adjustment is calculated till back-projection errors convergence. The initial 2D-to-3D correspondence is refined by the Bundle Adjustment scheme, and it is used to compute the final camera pose. The 3D point-cloud scene reconstruction is carried out, using all 2D camera views and 3D camera-specific information. A typical Bundle Adjustment output, is as shown in figure 6.

## 6. 3D Scene Surface Reconstruction

So far we have constructed a sparse 3D point-cloud of the viewed scene and to simultaneously recover camera poses, as shown in figure 6. In order to get a dense matching between multiple view images, we have combined two methods for this purpose. In order to get dense 3D oriented points, we combine two multi-view stereo matching methods: Iteratively-grown surface from matched points (Goesele et. al., 2007) and Patch-based Multi-view Stereo (Furukawa and Ponde 2007, 2009). After, the dense matching is done, we solve for the iso-surface solution in a Poisson surface reconstruction algorithm as given by Kazhdan et. al. (2006).

### 6.1.- Multi-view stereo matching

a) Iteratively grows surface from matched points (Goesele et. al., 2007): This stereo matching technique takes as input sparse 3D points reconstructed from bundle adjustment method and iteratively grows surfaces from these points. The main algorithm looks at both depth and normal vector, starting from an initial SIFT matched points or copied from previously computed neighbors. In order to find optimal surfaces around SIFT matches, we apply a two-level view selection camera: i) at the image level, global view selection identifies for each reference view a set of good neighborhood images to use for stereo matching. ii) at the pixel level, local view selection determines a subset of these images that yields a stable stereo match. The optimization stage for surface normals is within a photometry consistency calculation what significantly improves the matching results. We include reflectance coefficients between pair images in order to model changes in illumination. After we have run this stage, we have oriented points around SIFT matched points within a list of associated cameras, each potentially with different characteristics.

b) Patch-based Multi-view Stereo, PMVS (Furukawa and Ponce 2007, 2009): The patch-based multi-view stereo matching is complementary to the above stereo matching technique, i.e. we obtain additional oriented points, where the other method fails. PMVS is implemented as a match, expand and filter procedure, starting from a sparse set of SIFT matched points, and repeatedly expanding these to nearby pixel correspondences before using visibility constraints to filter away false matches:

- Matching: Features found by Harris and Difference-of-Gaussians operators are matched across multiple images, yielding a sparse set of oriented patches associated with salient image regions. Given these initial matches, the following two steps are repeated 3 times.
- Expansion: The technique is used to spread the initial matches to nearby pixels and obtain a dense set of oriented patches.
- Filtering: Visibility constraints are used to eliminate incorrect matches lying either in front or behind the observed surface.

The two methods yield a table of 3D-oriented points with a list of associated cameras. If we merged the two previous algorithms, we have a better representation of the 3D scene reconstruction. Now, we use a Poisson reconstruction algorithm in order to get the final 3D mesh associated with the final reconstructed model.

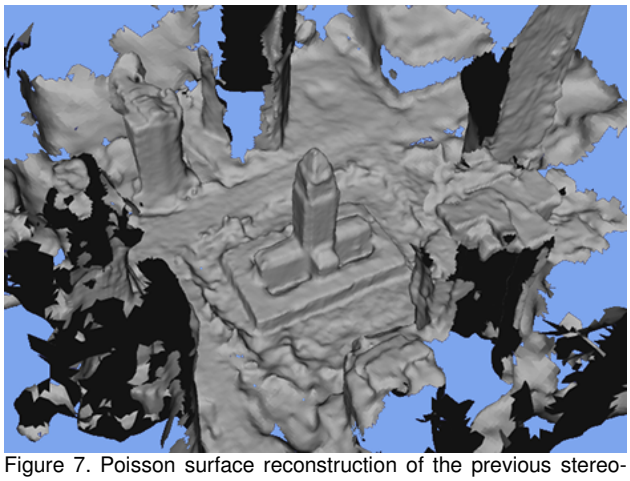


Figure 7. Poisson surface reconstruction of the previous stereo-matched mesh.

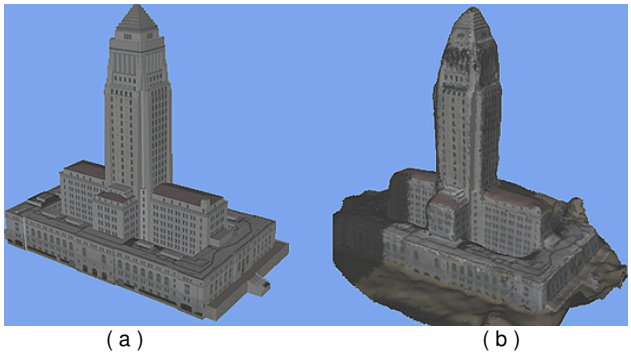
### 6.2.- Poisson Surface Reconstruction.

We follow the work done by Kazhdan et. al. (2006). We assume the surface reconstruction from oriented points can be cast as a spatial Poisson problem. This Poisson formulation considers all the points at once, without resorting to heuristic spatial partitioning or blending, and is therefore highly resilient to data noise. Unlike radial basis function schemes, the Poisson approach allows a hierarchy of locally supported basis functions, and therefore the solution reduces to a well-conditioned sparse linear system. Poisson surface reconstructions generate a mesh from a set of surface samples. After Poisson reconstruction is performed for the previous stereo matching step, we get a refined surface. Figure 7 shows the surface obtained from

the 3D-oriented points given by the merged stereo matches techniques above.

### 7. Textured 3D reconstructed scene

The texture is taken from the camera views and combined to form a multi-texture surface. In order to take into account the appropriate texture of the 3D reconstructed model, we follow two steps: i) Each reconstructed 3D point has a list of associated cameras, given by the SIFT matching process and the bundle adjustment process and ii) each 3D reconstructed point has a list of visibility after the stereo matching process is done. Thus, we have at least two criteria for the visibility for a given 3D reconstructed point. Since the stereo matching process has an implicit high degree of photometric consistency, we combine visibility information to take the camera that has the maximum, color cross-correlation within the cameras list in a small oriented patch around each 3D reconstructed point. We check RGB colour continuity around each orientated patch in order to assign texture on it. Figure 8 shows the textured 3D reconstructed model directly compared to the original CAD model.



( a ) ( b )

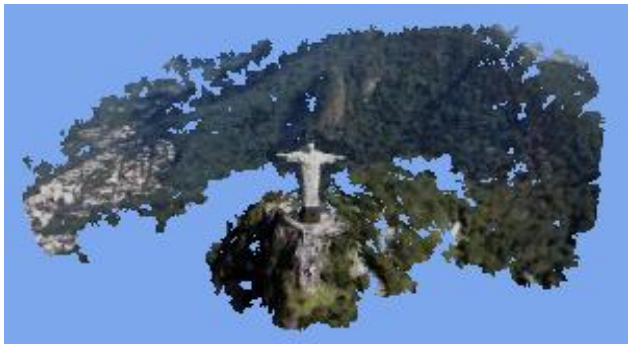
Figure 8. Original CAD building (a) compared to its 3D reconstruction from FMV (b).

### 8. Reconstructed terrain from real data

Beyond that we have tested our algorithm using synthetic images generated by ***Raphael GCS*** system. The main goal of our research is to get 3D reconstructed model from real data. In order to do so, real data videos can be downloaded from the popular website http://www.youtube.com. Consider that both real videos and photographs are taken under entirely separate conditions, such as aerial images from helicopters, full-moving videos with distinct resolution degree and noise level, dark and clear weather conditions, etc.

Therefore, we collected several videos of the Christ the Redeemer, which is a statue of Jesus Christ in Rio de Janeiro, Brazil. Our goal is to reconstruct the terrain around the peak of 700 meters Corcovado mountain in the Tijuca Forest National Park. On the other hand, the popular service provided by Google Earth does not show such statue at given coordinates 22º 57' 5.52" S, 43º 12' 39.32" W. Our 3D reconstructed system is ideal to completeness propose and beyond. The figure 9 shows four typical image of such data image set (a) and the final reconstructed terrain is shown in (b).

( a )



( b )

Figure 9. A practical application taken from youtube videos of the Christ the Redeemer in Rio de Janeiro, Brazil, where we can see sample views (a) and a snapshot of the reconstructed terrain (b).

### 9. Alignment between independent mission tracks.

We realise that, taking into account independent mission tracks under distinct operational conditions, we have additional information to get a more detailed 3D reconstruction, since the new images can be seen from different points of view and different illumination conditions, even using distinct imaging devices. This fact allowed us to write down an extra module that aligns independent 3D terrain reconstructions given by distinct mission tracks and

it gets better detailed 3D reconstruction with appropriate texture with reduced illumination effects.

In order to overlap more than two 3D oriented point sets, we have used a modified version of the Iterative Closest Point (ICP) algorithm developed by Besl and McKay (1992), Chen and Medioni (1992) and Zhang (1994). A classical ICP is based on the search for pairs of nearest points in two data sets and estimates the rigid body transformation that aligns them. The rigid body transformation is then applied to the points of one set and the procedure is iterated until convergence is achieved. This point-to-point scheme has low convergence. Thus, we have used a modified version of the ICP, the point-to-tangent plane approach originally proposed by Potmesil(1983).

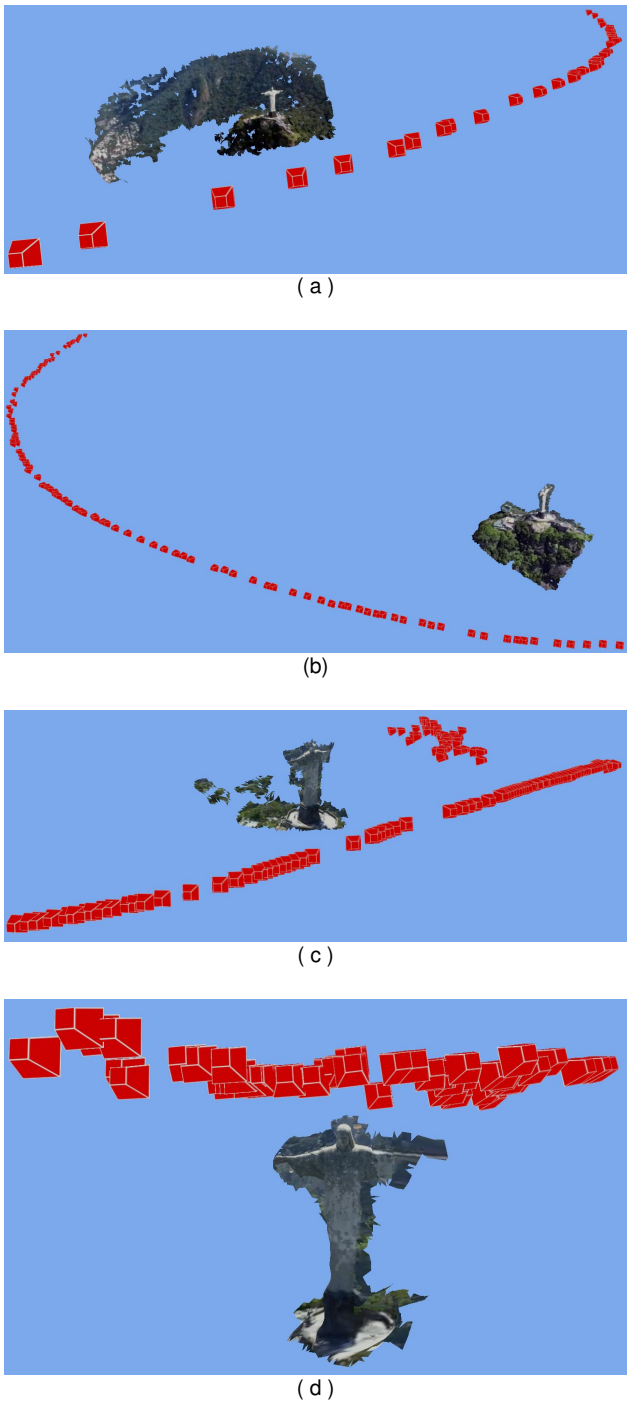
The search for pairs of nearest points in two data sets, is done using a kd-trees structure. Thus, the Approximate Nearest Neighbours package of Arya, et. al. (1998) is used extensively in this process. On the other hand, the rigid body transformation is given by a translation vector *T(x, y, z)* and a matrix of rotation *R(ω, φ, κ)*. We also include a scale factor *S,* given that we want overlap two mesh surfaces derived from distinct camera model. We can do this without affecting the whole reconstruction. In order to find the correct rigid body transformation, we have used a Levenberg-Marquardt algorithm as the least square method to solve for the translation vector, rotation matrix and scale factor. Furthermore, we have included in the least square problem information about normal vectors over each matched pair point and its texture (colour).

Finally, we get an enlarged list of 3D orientated points with additional information about mixed cameras. This new list of 3D orientated points is filtered out using the Poisson reconstruction process to get a new 3D surface reconstructed model with additional detail given by the combination of multiple and independent mission tracks. Since we have more 3D oriented points with associated camera-lists, we can get a better texture 3D surface model.

Of course, this extra images could include images taken from the ground, thus being complementary to those taken from satellites, LIDAR and unmanned vehicles in more than one mission and at different times in order to derive a high-quality 3D reconstruction of any target of interest.

**9.1 Christ the Redeemer derived from independent mission tracks.**

The trajectories considered to get an accurate 3D terrain reconstruction around the Christ the Redeemer are shown in figure 10.



( a )

(b)

( c )

( d )

Figure 10.- It shows three trajectories to reconstructed a more accurate 3D terrain model.

Figure 10 (a) is a reconstruction calculated from a video with 320x240 pixels size with 6 seconds length. Figure 10 (b) shows a reconstruction calculated from a video with 254x271 pixels size with 18 seconds length. Figures 10 (c) and (d) show a reconstruction derived from a video with 343x259 pixels size with 35 seconds length.

### 10. Comparison between CAD model and 3D reconstructed scene.

We have tested this algorithm on a number of real-life objects but here we show a study of a building of known dimensions in order to quantitatively estimate the algorithm's accuracy. We have quantified our 3D reconstruction process through a typical 3D scene reconstruction with a well-known CAD model. This is done within the synthetic vision module of ***Raphael GCS*** in order to compare the 3D surface reconstruction and the original 3D mesh model point set. We have taken the template and fixed and fitted the 3D reconstructed mesh using a modified version of the ICP algorithm as above. We search in an iterative process for a rigid body transformation that includes normal vector information and texture (colour) between this two point sets, and we get the final overlapping mesh between template and 3D reconstructed model.

In Figure 11 we show the overlap between the 3D reconstructed model (red dots) and the its well-known CAD model (blue dots). We read in the CAD model from an .OBJ file. As we can see the agreement is excellent.

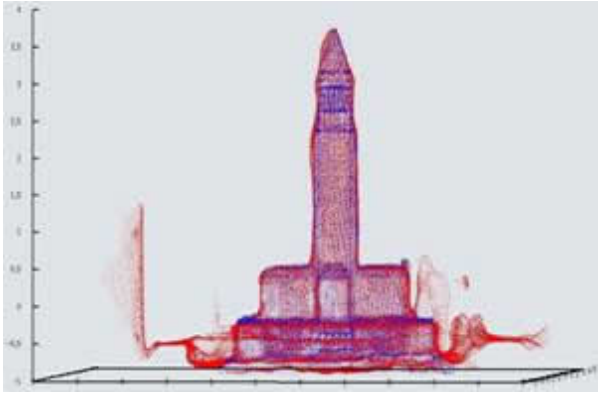


Figure 11. The alignment between an oriented 3D reconstructed mesh (red dots) and the CAD model mesh (blue dots).

Thus, we calculated a histogram of error associated over each axis. Figure 12 shows a histogram of the ratio between the shortest distance between both model and 3D reconstruction divide by a characteristic scale, in this case the building height. The average standard deviation about the three axes is around 1.0%.

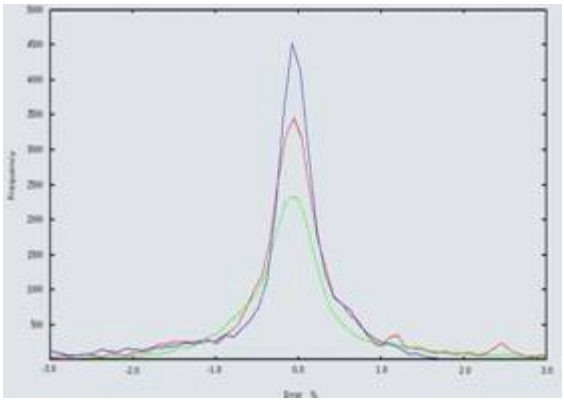


Figure 12. Histogram of error about the Z-axis (blue line), Y-axis (red line) and X-axis (green line). Standard deviation = 1.0%.

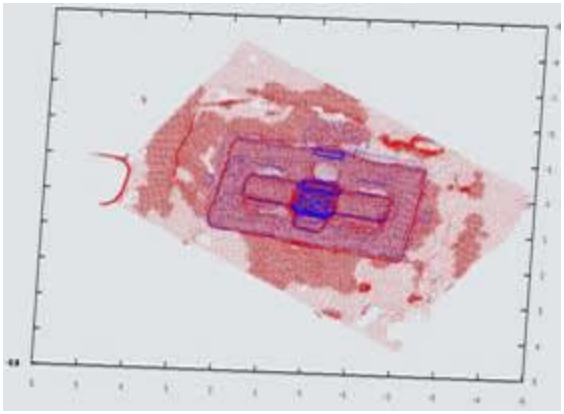


Figure 13. It shows same as above, but it seen from top-to-bottom.

### Conclusions

We have produced an efficient software system capable of recovering 3D terrain information from any combination of full-moving video and still imagery, even when taken under varying illumination conditions and by a range of imaging systems. It is a fundamental need to attention to the camera modelling, its effects and correction coefficients in order to obtain accurate 3D reconstructions. The illumination, scale and rotation effects is overcome by including them into a database by the SIFT methods on the correspondence between multiple view images.

Computer vision algorithms are used to reconstructed camera trajectory and pose. We have combined a stereo matching algorithm to get 3D oriented point sets and then, we applied a Poisson

reconstruction algorithm to get a refined 3D mesh. Additionally, we have designed a module that aligns two o more 3D reconstructed meshes given by distinct mission tracks. We verified the system by designing a module that compares CAD models with their 3D reconstructed estimates, and found the system to yield errors of the order of 1%.

## References


[1] Arya S., Mount D. M., Netanyahu N. S., Silverman R., Wu A. Y. 1998. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. of the ACM 45*, 6, 891–923.

[2] Besl P. J., and McKay N. D., 1992. A method for registration of 3D shapes. *IEEE PAMI*, 14(2), 239-256.

[3] Chen Y., and Medioni G., 1992. Object modelling by registration of multiple range images. *IVC*, 10(3), 145-155.

[4] Fischler M., Bolles R. 1987. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Readings in computer vision: issues, problems, principles, and paradigms,* 726–740.

[5] Furukawa, Y., Ponce, J. 2007. Accurate, Dense, and Robust Multi-View Stereopsis. *Computer Vision and Pattern Recognition, 2007. CVPR 07.* 1-8.

[6] Furukawa, Y., Ponce, J. 2009. Accurate Camera Calibration from Multi-View Stereo and Bundle Adjustment Source. *International Journal of Computer Vision 84*, 257-268.

[7] Goesele, M. Snavely, N. Curless, B. Hoppe, H. Seitz, S.M. 2007. Multi-View Stereo for Community Photo Collections. *Computer Vision, 2007. ICCV 2007*, 1-8.

[8] Gutiérrez-Resendiz F. and Funes-Gallanzi M., 26 May 2009. "*Raphael Ground Control and Post-Processing Station -General Description-*". Available online at http://www.avntk.com/files/Raphael.pdf

[9] Hartley R. I., Zisserman A. 2004. Multiple View Geometry. *Cambridge University Press*, Cambridge, UK.

[10] Heikkila J., and Silvén O., 1997. A four-step camera calibration procedure with implicit image correction. Infotech Oulu and Department of Electrical Engineering. University of Oulu. Finland.

[11] Kazhdan M., Bolitho M., Hoppe H., 2006. Poisson Surface Reconstruction. *Eurographics symposium on Geometry processing 2006.* 61-70.

[12] Lourakis M. I. A., Argyros A. A., 2004. The design and implementation of a generic sparse bundle adjustment software package based on the Levenberg-Marquardt algorithm. *Institute of Computer Science - FORTH*. 340.

[13] Lowe D. 2004. Distinctive image features from scale-invariant keypoints. *Int. J. of Computer Vision 60*, 2, 91–110.

[14] Potmesil M., 1983. Generating models of solid objects by matching 3D surface segments. International Joint Conference on Artificial Intelligence. *Karlsruhe*, 1098-1093.

[15] Zhang Z., 1994. Iterative point matching for registration of free-from curves and surfaces. *Int. Journal of Computer Vision*, 13(2), 119-152.