Independent Study on 3-D Reconstruction and Related Problems in Computer Vision

Anant Jain*

Supervised by: Dr. Subhasis Banerjee†

May 9th, 2011

Abstract

This report is a compendium of the work done as a part of an Independent Study on 3-D Reconstruction and other topics in Computer Vision in conjunction with CSL840 lecture course. The report has brief descriptions and compilation of results of the assignments, lectures, exams and a final project on geotagging web images using (pre-)computed sparse point cloud models of landmarks. The topics covered over the lectures included the problem of 3-D reconstruction - single view geometry, multiple view geometry and affine multiple view geometry, robust computation techniques, Scale Invariant Feature Transform, Pyramid methods in Image Processing, Graph Cut methods, Viola Jones method of face detection, problem of classification, Lucas Kanade algorithm and the KLT and Kalman trackers for alignment problems and a few other miscellanous topics. Part I covers the assignment on removal of affine and projective transformative effects for measurement of distances on a plane in a single view. Part II covers the assignment on various techniques tried out for camera calibration including the industry standart Zhang calibration on OpenCV. Part III outlines the work done towards the final project. Part IV is a compilation of the Minor and Major papers.

*2008EE10330, Department of Electrical Engineering, Indian Institute of Technology, Delhi

†Department of Computer Science and Engineering, Indian Institute of Technology, Delhi. www.cse.iitd.ac.in/~suban

Contents

List of Figures

Part I

Removal of Affine and Projective Transformations for measurement of distances on a plane in a single view.

1 Problem introduction:

A projective transformation can be decomposed into a chain of transformations (Euclidean, affine, projective):

$$\mathbf{H} = \mathbf{H_E H_A H_P} = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0^T} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0^T} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{v^T} & v \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{tv} \\ \mathbf{v^T} & v \end{bmatrix}$$

The decomposition is valid if $v \neq 0$ and unique if s is positive.

The goal of this part is to be able to remove projective transformations to get a Euclidean reconstruction of a plane from a single view, so as to make measurements on that plane. Two approaches are considered for this assignment. The first one is a stratified approach, wherein first projective distortions are removed followed by affine distortions. The second one is a one step method given by Hartley and Zisserman (pages 35,37.) Both the approaches are discussed below.

2 Approach 1: Stratified approach

2.1 Step 1: Affine rectification

Since parallel lines remain parallel under affine distortion, we can recover the affine properties from images by the transformation matrix $\mathbf{H}$ that maps the vanishing line back into the line at infinity. If the imaged line at inifinity is $l = (l_1, l_2, l_3)^T$, then provided $l_3 \neq 0$, a suitable projective transformation that maps l back to l_∞ is

$$\mathbf{H} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{l_1} & \mathbf{l_2} & \mathbf{l_3} \end{bmatrix} \mathbf{H_A}$$

2.2 Step 2: Metric rectification from affine image

Now that we have the affinely rectified image, we want to find the affine transform matrix

$$\mathbf{H} = \begin{bmatrix} A & t \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$$

It is shown in Hartley and Zisserman's book [2] that enforcing orthogonality between two pairs of orthogonal lines are enough to solve for A and t and hence **H.**

3 Approach 2: Metric rectification via the estimation of the dual conic C^*_∞

If the point transformation from the world to the image is $x' = Hx$ where the x frame is Euclidean and the x' frame is projective, C^*_∞ transforms as $C^*_\infty = HC^*_\infty H^T$. Once C^*_∞ has been identified on the projective plane, the rectifying homography can be directly estimated. Writing C^*_∞ as (use SVD):

$$\mathbf{C}^*_\infty = \mathbf{U} \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{U^T}$$

we get the rectifying homography as $H^{-1} = U^{-1}$. Hartley and Zisserman[2] suggest that we determine C^*_∞ on the perspectively imaged plane by specifying five orthogonal line pairs and fitting a conic (using the constraint $l^T C^*_\infty r = 0$).

4 Results

The stratified approach was implemented successfully in MATLAB. The user is asked to specify a set of four vertices in a rectange (a tile) for the affine rectification.

For the metric rectification, the user is asked to specify two rectangles (tiles) in clockwise order so that two pairs of orthogonal lines can be derived from them to fit the dual conic C^*_∞, and hence get the metric rectification by getting the appropriate corrective homography H.

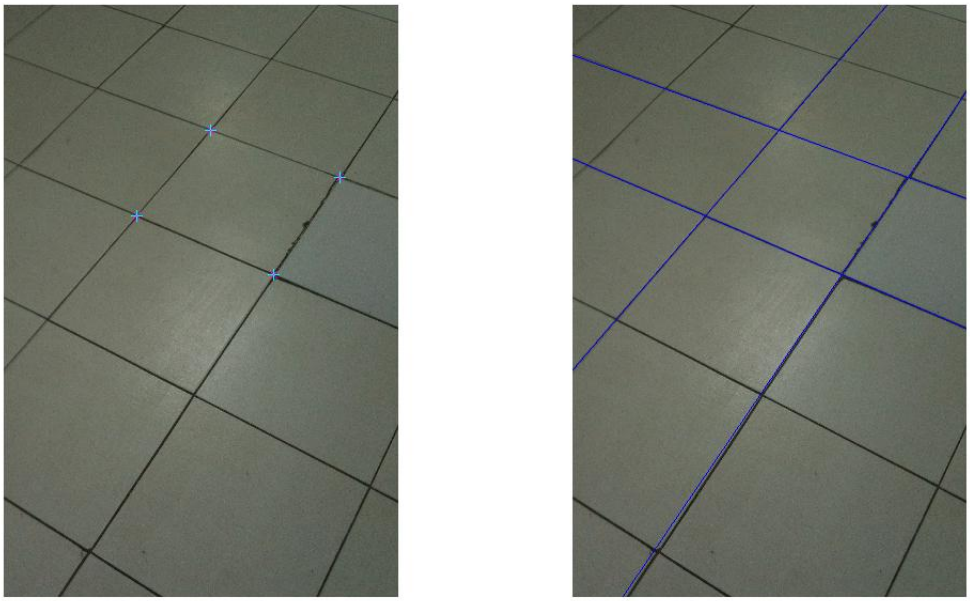

Figure 1: Picking points on the image, lines fit through the points.

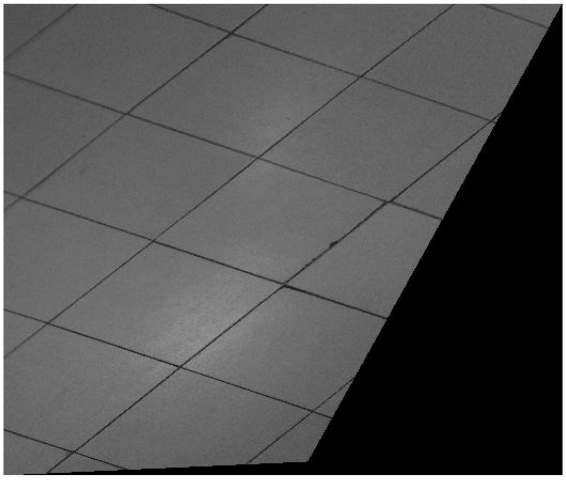

Figure 2: After Affine rectification

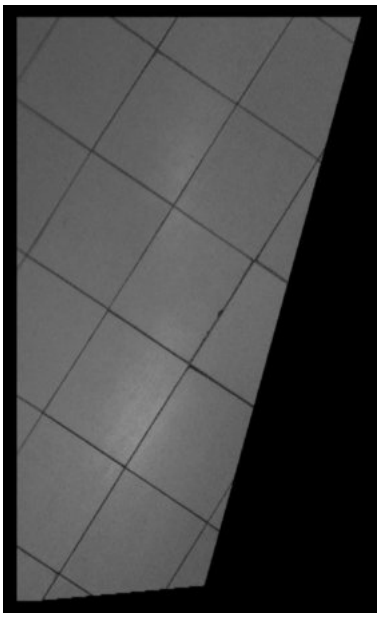

Figure 3: After Metric Rectification

Part II
Camera Calibration techniques

5 Introduction to Camera Calibration

Camera calibration is the process of finding the true parameters of the camera that produced a given photograph or video. Usually, the camera parameters are represented in a 3×4 matrix called the camera matrix. It is convenient to break down the camera matrix into two sets of parameters: external and internal, i.e. $\mathbf{P_E} = \mathbf{C}[\mathbf{R}|\mathbf{t}]$ where $\mathbf{C}$ is the matrix of camera internals and $[\mathbf{R}|\mathbf{t}]$ is the matrix of camera externals.

6 Calibration using three planes by finding C from image of absolute conic.

The image of three squares on three different planes (not necessarily orthogonal) are sufficient to give calibration. Consider the following steps:

1. For each square compute the homography $\mathbf{H}$ that maps its corner points to their imaged points.

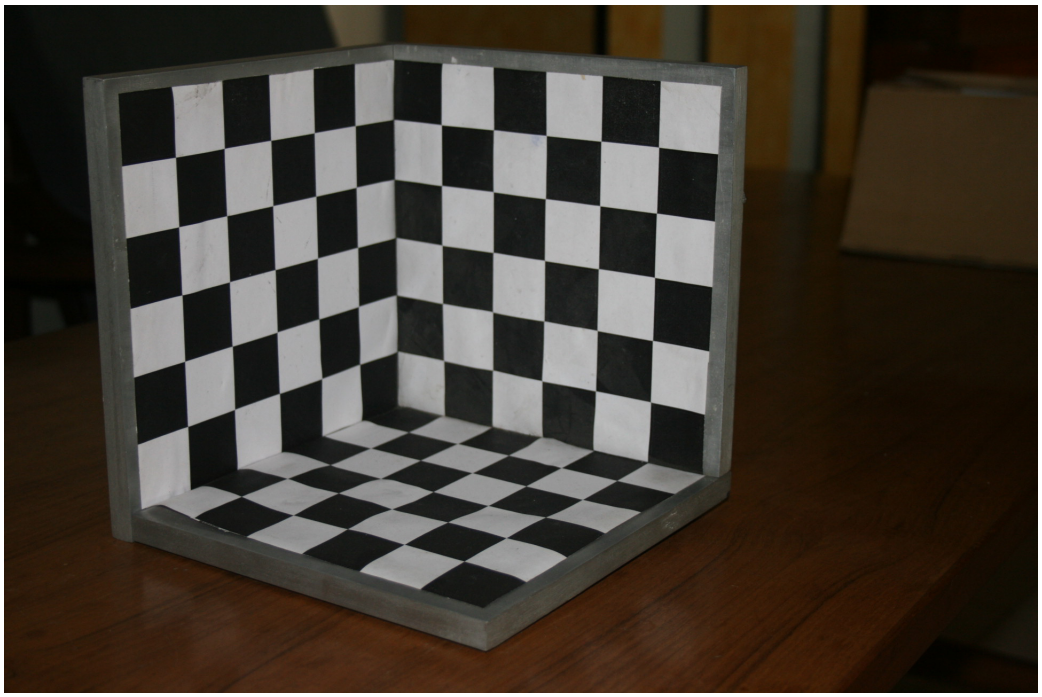

Figure 4: Image used for camera calibration

2. Compute the imaged circular points for the plane of that square as $\mathbf{H}(1, \pm i, 0)^T$.

3. Fit a conic ω through the six imaged points. Note that five points are sufficient to define a conic.

4. Compute $\mathbf{C}$ from $\omega = (\mathbf{C}\mathbf{C}^{\mathbf{T}})^{-1}$ using Cholesky decomposition.

A simple MATLAB code gave the following camera internals for the image in figure 4

$$\mathbf{C} = \begin{bmatrix} \mathbf{3109.7} & \mathbf{100.2} & 0.3 \\ 0 & 3169.0 & 0.2 \\ 0 & 0 & 1.0 \end{bmatrix}$$

7 Zhang's calibration

Zhang's calibration is the method implemented in OpenCV's cameracalibrate2 routine. The theory for this method was covered in class and can be found in lecture notes on the course webpage [3]. The resulting camera intrinsics for the

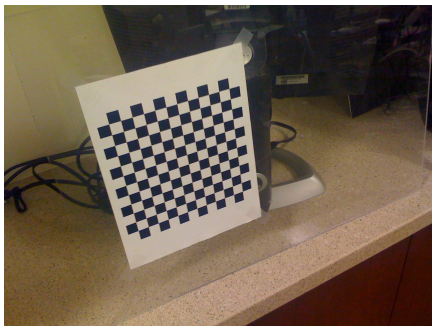

Figure 5: Image used for Zhang's calibration

object used in Figure 5 were found to be

$$\mathbf{C} = \begin{bmatrix} \mathbf{1764.3} & \mathbf{0.819} & 0.509 \\ 0 & 1766.1 & 627.96 \\ 0 & 0 & 1.0 \end{bmatrix}$$

which look reasonable. The camera distortion parameters are found as well.

Part III
Project: Geotagging web images using (pre-computed) sparse point cloud models of landmarks.

8 Problem Introduction

Geo-tagging is a fast-emerging trend in digital photography and community photo sharing. The presence of geographically relevant metadata with images and videos has opened up interesting research avenues within the multimedia and computer vision domains. The idea of the project is to take an image and the name of the landmark visible in the image, and output the geotag for the image, where geotag is the tuple $\{latitude, longitude, accuracy\}$. A related work [4] by Roberto Cipolla et al. was reviewed by us. Their approach is based on an LVSM done on visual vocabulary constructed using SIFT features. Our work attempts to use the the bundler tool[5] made available by Noah Snavely to the public domain to get a sparse point reconstruction of the landmark, thereby getting the camera centres for all the images. The approach is explained in the next section.

9 Pipeline and algorithms

Figure 7 outlines the pipeline used.

The first step of the pipeline is to construct the 3-D model of the landmark if not already present in the database. The process for this sparse point 3-D model construction is as follows: First download a first few images from google images. This number is determined by the number of images the bundler can process in a reasonable amount of time. In our tests, this number was taken to be of the order of 100. The next step is to run the bundler on this set of images. The following steps summarize the bundler:

1. Extract EXIF tags to read the focal length (additionally, store the geotags too if available.)
2. Convert to grayscale .pgm and extract SIFT features.
3. Use ANN Library to match the SIFT features.
4. RANSAC-based robust estimation of fundamental matrix depending of availability of focal length for verification.
5. In our model, we assume that Google has taken care of the methods employed for selecting the best bucket to be sent in the bundler for building Rome in a day, namely:

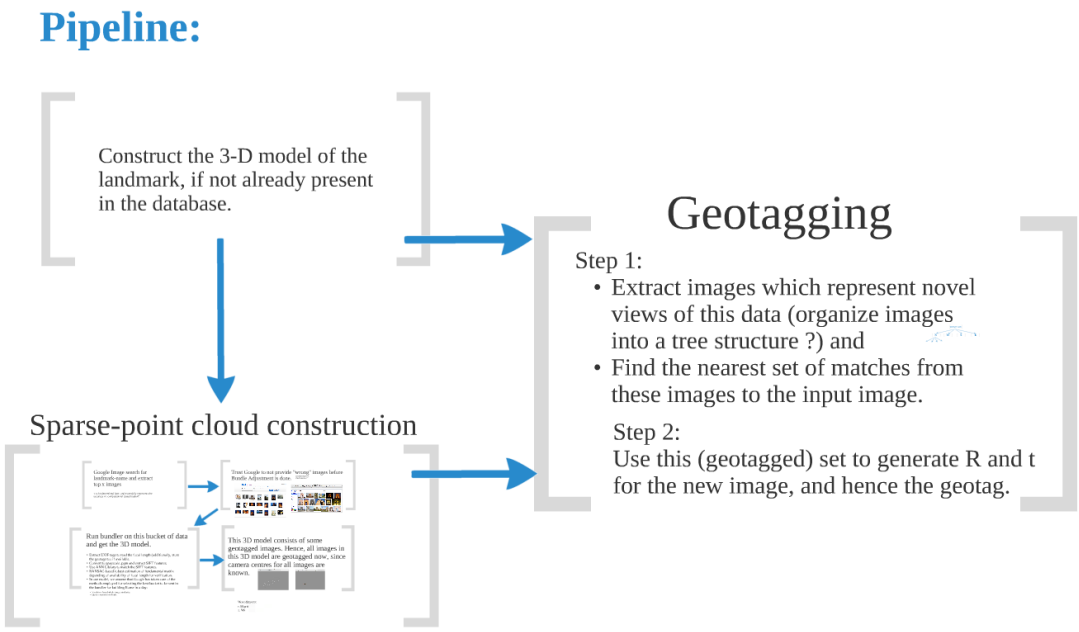

Figure 6: Pipeline for geotagging

(a) Vocab tree based whole image similarity.

(b) Query expansion methods.

The output of the bundler gives us the camera centres of all the images fed into the bundler. This 3D model consists of some geotagged images. Hence, all images in this 3D model are geotagged now, since camera centres for all images are known. The last step is to geotag an image, given the hence constructed 3-D model. The following steps outline the procedure:

1. Step 1: Extract images which represent novel views of this data (organize images into a tree structure ?) and Find the nearest set of matches from these images to the input image.

2. Step 2: Use this (geotagged) set to generate R and t for the new image, and hence the geotag.

This completes the description of the pipeline. The results are described in the next section.

10 Results

3-D reconstructions were done for India Gate situated in New Delhi. Top 150 image search results were collected both from Flickr and Google, but the bundler practically failed on Flickr images, owing to a considerable number of wrongly tagged images. Two artificial datasets were also constructed: of the Bharti building and Multi Storey (MS) building of IIT Delhi. Some representational images of the three datasets are given in Figure 7. The output of bundler for India Gatem Bharti and MS is shown in Figures 8, 9 and 10 respectively.

The inaccuracy of giving a new geotag is less than 1 m, which is far better than that available on commercial high-end cameras and mobile phones. (5-10 m)

Figure 11 shows the tags for a few untagged photographs after geotagging is completed using the bundler output. Note that the least count of the geotag, i.e. of $(latitude, longitude)$ pair results in apparently grid like appearance of the tags.

11 Conclusions:

The approach to geotag images by making use of a sparse point cloud model of landmarks is unique. The problem deserves further exploration and rigorous experimentation and implementation to create a robust service for geotagging images, which can be further deployed in multiple applications - most importantly, in creating Virual Reality models of cities.

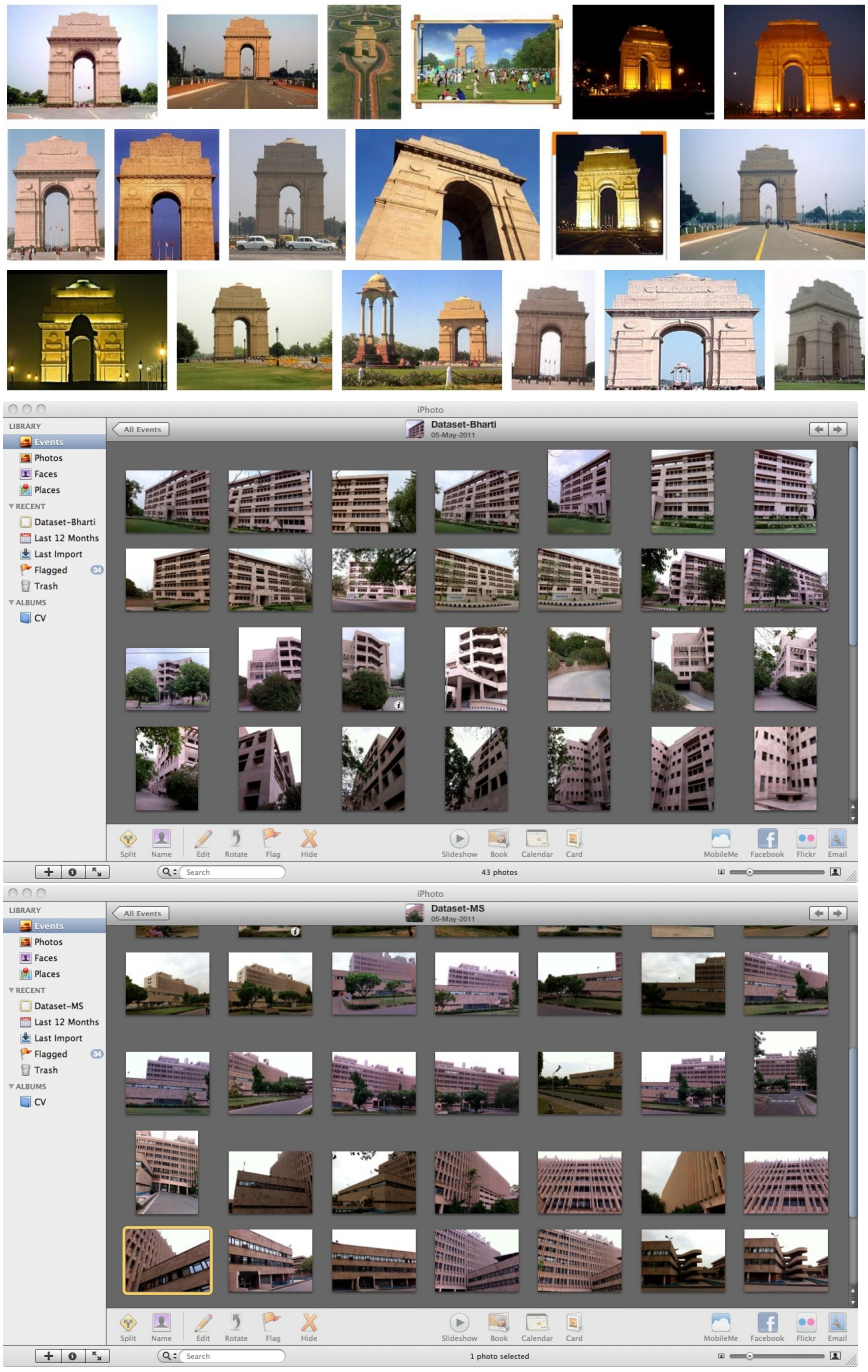

Figure 7: Datasets used for the evaluation of the algorithm: IndiaGate, Bharti and MS

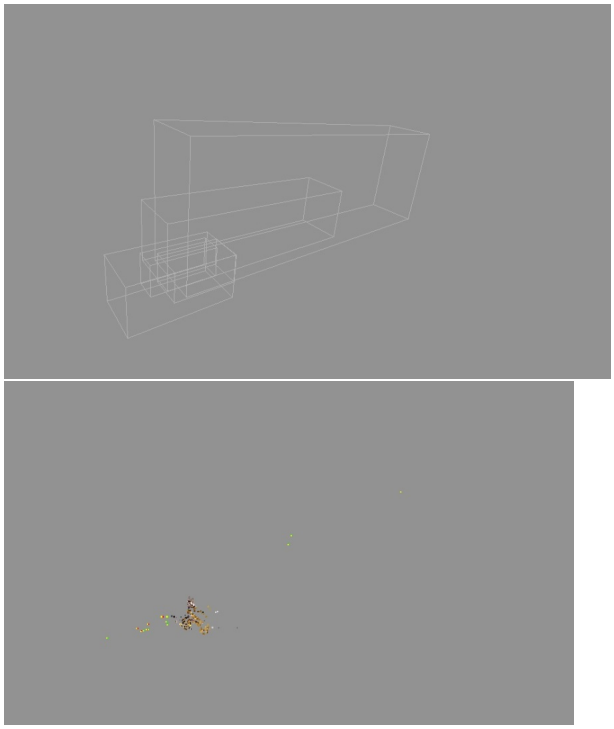

Figure 8: Bundler output for Indiagate dataset

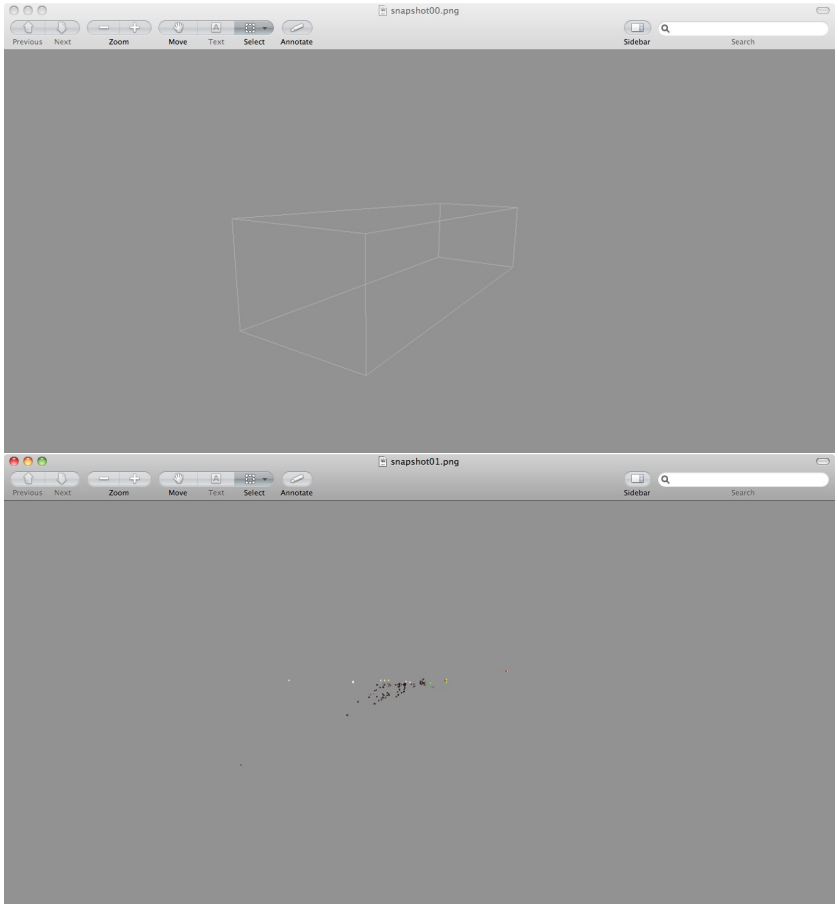

Figure 9: Bundler output for Bharti dataset

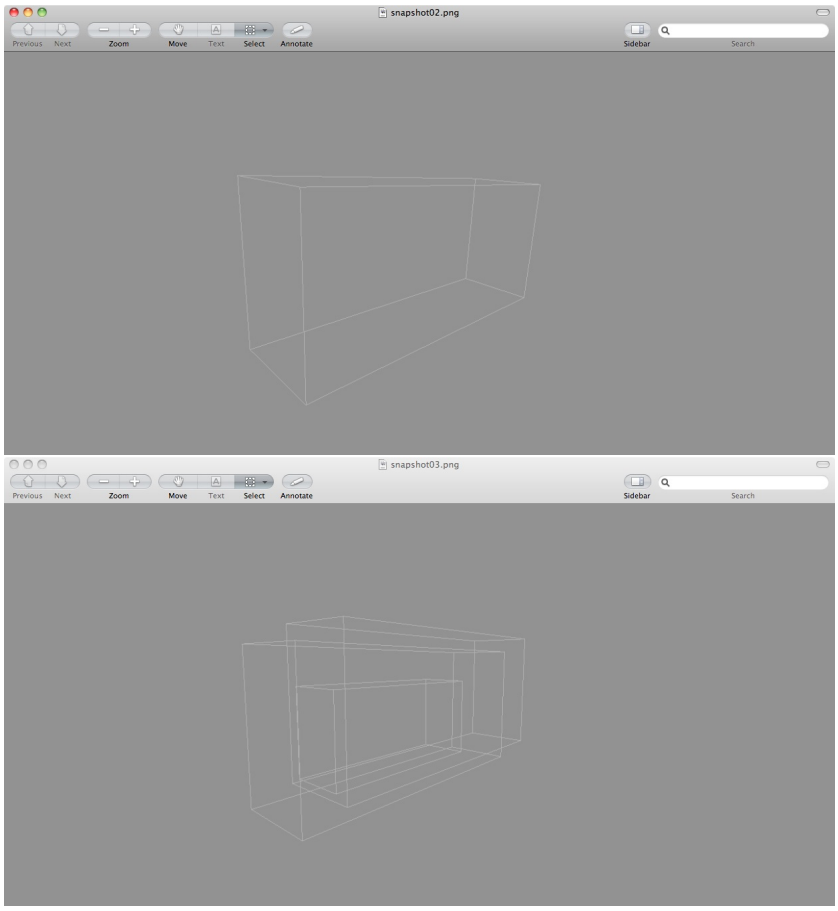

Figure 10: Bundler output for MS dataset

Figure 11: Geotags for untagged images near Bharti

References

[1] A. Criminisi, I. Reid, and A. Zisserman. Single view metrology. IJCV, 40(2):123–148, 2000.

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

[3] Subhasis Banerjee. Projective geometry, camera models and calibration, `http://www.cse.iitd.ac.in/~suban/vision/geometry/geometry.html`

[4] Junqiu Wang, Student Member, IEEE, Hongbin Zha, and Roberto Cipolla, Member, IEEE. Coarse-to-Fine Vision-Based Localization by Indexing Scale-Invariant Features

[5] Noah Snavely, Bundler: Structure from Motion (SfM) for Unordered Image Collections. `http://phototour.cs.washington.edu/bundler/`

Part IV
Solutions to exams:

Minor-1 Report

CSL840

Submitted by: Anant Jain, 2008EE10330

Degree of Originality: This is to ascertain that this assignment has been done solitarily by me, and no discussions contributed to this assignment.

1. Explanations:

   (a) A potrait can be considered to be a two dimensional image of a 3 dimensional object. Since the image of this two dimensional image on a flat canvas would look the same from every angle (modulo some affine stretching,) an image of a potrait looking straight outwards would be perceived to be looking straight outwards towards the viewer from every angle. Thus, a potrait's eyes seem to follow the subject around the room.

   (b) Infinitely distant objects lie on the plane at infinity, π_∞ . A point on this plane can be represented as $(d^T, 0)$. Let the image of such a point be x. Thus x is given as:

   $$x = P(d^T, 0) = CR[I|t](d^T, 0) = CRd$$

   Thus, the planar homography between π_∞ and the image plane doesn't involve the translation term, t. Hence, images of "infinitely" distant objects like stars and the moon stay fixed on your retina as you translate, but change with rotation.

2. Mathematics Building question:

   (a) Single-view analysis:

      i. **Affine measurements on any one of the faces of the 'Mathematics' building.**

         A. Considering the frontal face of the mathematics building a planar surface is a reasonable assumption. An affine correction of this plane shall require estimation of its line at infinity. This line at infinity is obtained as the line joining (minimum) two vanishing points of any two (or more, for robust computations) parallel lines on this plane. If the imaged line at inifinity is $\mathbf{l} = (l_1, l_2, l_3)^T$, then provided $l_3 \neq 0$,

a suitable projective transformation that maps $\mathbf{l}$ back to $\mathbf{l}_\infty$ is given by:

$$\mathbf{H} = \begin{bmatrix} \mathbf{1} & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{bmatrix} H$$

ii. **Euclidean measurements on any one of the faces of the 'Mathematics' building.**

A. There are two methods for doing metric correction of a plane. **First method:** start with the Affine correction done in part (i). The Euclidean correction would need specification of a measurement in any two non-parallel directions on the plane of the mathematics building as well as measurement of the angle between any two (non-parallel) directions. Essentially, we are using the idea of decomposition of a projective transformation into a series of Euclidean, Affine and Projective transformations, wherein, after affine correction, the effect of the last Projective has been removed, and we need to find $\mathbf{H_A^{-1}} = \begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0^T} & 1 \end{bmatrix}^{-1}$ where, $\mathbf{B}$ is an upper triangular matrix (3x3) normalized as $det(\mathbf{B}) = 1$. **Second method:** can be the same as the method of metric rectification given by Hartley and Zisserman via the estimation of the dual conic (and discussed in Q5) This method requires specification of (atleast) five pairs of perpendicular set of lines on the plane.

iii. **Ratio of the heights of the 'Workshop' and the 'Mathematics' buildings.**

A. This one is answered on the basis of section 2.2: Measurements on parallel planes of [**?**]. We can compare measurements made on two separate planes by mapping between the planes in the reference direction via the homology. A map in the world between parallel planes induces a map in the image between images of points on the two planes. This image map is a planar homology, which is a plane projective transformation with five degrees of freedom, having a line of fixed points, called the axis and a distinct fixed point not on the axis known as the vertex. Planar homologies arise naturally in an image when two planes related by a perspectivity in 3-space are imaged. In particular we may compute (i) the ratio between two parallel lengths, one length on each plane; (ii) the ratio between two areas, one area on each plane. In fact we can simply transfer all points from one plane to the reference plane using the homology and then, since the reference plane's vanishing line is known, make affine measurements in the plane, e.g. parallel length or area ratios.

Figure 12: Measuring distance between Maths department and workshop

iv. **The absolute location and height of the 'Workshop' building in Euclidean terms.**

A. Refer to section 2.1: Measurements between parallel planes of [**?**]. Figure 1 below shows the two parallel planes (the faces of Mathematics and workshop buildings) and a set of lines joining the two planes, and perpendicular to them, alongwith their vanishing point.

B. The method outlined there is reproduced here : Consider figure 2. We wish to measure the distance between two parallel planes, specified by the image points and, in the reference direction. Figure 2 shows the geometry, with points and in correspondence. The four points marked on the figure define a cross-ratio. The vanishing point is the image of a point at infinity in the scene. In the image the value of the cross-ratio provides an affine length ratio. In fact we obtain the ratio of the distance between the planes containing and, to the camera's distance from the plane (or depending on the ordering

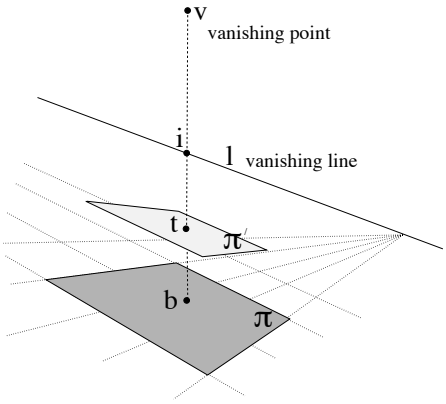

Figure 13: Figure 2 from [**?**]

of the cross-ratio). The absolute distance can be obtained from this distance ratio once the camera's distance from is specified. However it is usually more practical to determine the distance via a second measurement in the image, that of a known reference length.

C.

D. The absolute height of the workshop is determined from height of mathematics building found in part (ii) and the ratio of the heights of the two buildings found in part (iii).

v. **The camera internal calibration matrix K.**

A. As discussed in the class (and the notes,) all we need to specify for the internal calibration matrix are the vanishing line and one vanishing point in a direction not lying on the plane. These can be found easily in the same way as done in Figure 1 for one axis.

(b) **Estimation of zoom factor:**

i. Once an Euclidean construction has been done for both the images (a) and (b), we know easily find out the number of pixels spanning the same real length (say height of maths department.) The zoom factor can be estimated by the ratio of this quantity for the two cases.

(c) **Estimation of rotation axis and rotation angle:**

i. Consider two images of a scene obtained by the same camera from different position and orientation. The images of the points at infinity, the vanishing points, are not affected by the camera translation, but are affected only by the camera rotation R.

ii. Consider a scene line with vanishing point v in the first view and v' in the second. The vanishing point v has a direction d in the first cameras Euclidean frame, and, similarly, the vanishing point v' has a direction d' in the second cameras Euclidean frame. We have

$$d = C^{-1}v/||C^{-1}v||$$
$$d' = C^{-1}v'/||C^{-1}v'||$$

iii. The directions are related by $d = Rd'$ which represents two independent constraints on R.

iv. Hence, the rotation matrix can be computed from two such corresponding directions provided we know C. This rotation matrix gives the rotation angle.

v. We already know that the rotation axis has to pass through the camera centre (already available from last column of P.) The direction of rotation axis is the same as the direction whose vanishing point doesn't change.

vi. A transformation to interpolate an in-between image between the two in terms of a rotation angle φ from the first image. If K, the camera calibration matrix is written as $C[R|t]$, then for the rotation angle φ we can find a rotation matrix R' such that $RR' = R''$. Thus $C[R"|t]K^{-1}$ serves as the transformation from the first image.

3. **Decomposition of projective transformation:**

(a) Start with product:

$$\mathbf{H_E H_A H_P} = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0^T} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0^T} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{v^T} & v \end{bmatrix} = \begin{bmatrix} s\mathbf{RB} & \mathbf{t} \\ \mathbf{0^T} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{v^T} & v \end{bmatrix}$$

$$= \begin{bmatrix} s\mathbf{RB} + \mathbf{tv^T} & \mathbf{tv} \\ \mathbf{v^T} & v \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{tv} \\ \mathbf{v^T} & v \end{bmatrix} = \mathbf{H}$$

Thus, $\mathbf{A} = s\mathbf{RB} + \mathbf{tv^T}$. If B is an upper triangular matrix, with det(B) = 1, $\mathbf{H_A}$ is an affine transformation.

i. $\mathbf{H_P}$ has 2 dof (last row has three elements, but we are concerned upto a scale only, hence only 2 dofs) and moves the line at infinity in the image, since the 2x2 submatrix is an Identity matrix.

ii. $\mathbf{H_A}$ has 2 dof and affects affine properties, but can't affect line at infinity since the last row is $(\mathbf{0^T 1})$

iii. $\mathbf{H_E}$ has 4 dof and is a general similarity transformation which doesn't affect affine (again, since last row is $(\mathbf{0^T 1})$) or projective properties.

(b) Reverse order of decomposition:

$$\mathbf{H_P H_A H_E} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{v^T} & v \end{bmatrix} \begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0^T} & 1 \end{bmatrix} \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0^T} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{v^T} & v \end{bmatrix} \begin{bmatrix} s\mathbf{RB} & \mathbf{Bt} \\ \mathbf{0^T} & 1 \end{bmatrix}$$

$$= \begin{bmatrix} s\mathbf{RB} & \mathbf{Bt} \\ \mathbf{v^T sRB} & \mathbf{v^T Bt}+v \end{bmatrix} = \begin{bmatrix} \mathbf{A} - \mathbf{tv^T} & \mathbf{Bt} \\ \mathbf{v^T sRB} & \mathbf{v^T Bt}+v \end{bmatrix} = \mathbf{H}$$

4. **Show that given two intervals on an imaged line with known length ratios the vanishing point of the line can be determined.**

   (a) The method for the same was read by me in *Multiple view geometry in computer vision by Hartley and Zisserman*, page 51, before attempting this paper, and is re-produced below:

      i. Given three points a', b', c' in the image, measure the ratio distance ratio in the image, d(a',b'):d(b',c') = p':q'

      ii. Let the measured (Euclidean) distance ratio of the corresponding world points a,b,c be p:q

      iii. These set of points (a,b,c) may be represented as $(0,1)^T, (p,1)^T, (p+q,1)^T$ in 1-D projective coordinates. The similar representation for 1-D image points (a',b',c') is $(0,1)^T, (p',1)^T, (p'+q',1)^T$.

      iv. Compute a 1-D homography $\mathbf{H_{2x2}}$ from world points (a,b,c) to image points (a',b',c'), and the image of $(1,0)^T$ under this homography gives the vanishing point of the line containing a',b',c'.

5. Solutions:

   (a) **Show that the constraint specified by (iv) can be deduced from (i), (ii) and (iii).**

      i. The horizontal lines in (i) and (iii) give the horizontal axis' vanishing point. The vertical lines in (i) and (ii) give the vertical axis vanishing point. Also, as can be seen visually, the choice of the the new constraint (iv) uses the same orthogonal relationship of the horizontal and vertical axis. Thus (iv) can be deduced from (i), (ii) and (iii).

   (b) **Can constraint (v) also be deduced from (i), (ii) and (iii)? If not, what extra information does it provide?**

      i. No, the constraint (v) cannot be deduced from (i), (ii) and (iii). It provides the orthogonality constraint in a different axis pair, though on the same plane.

   (c) **A student from a previous batch stacked up the five constraints in a 5 × 6 matrix for conic fitting (see [3] - page 9). He claimed that the matrix is of rank 5. Is this possible in view of the fact that one or more of the constraints can be deduced from the others? Did he do some gadbadi?**

i. No, he didn't do any*gadbadi*. The rank of the matrix is still 5 due to the fact that the derivation of constraint (iv) from constraints (i), (ii) and (iii) was dependent on the fact that we started with the knowledge that we were working in the same plane. This need not be true for a general case, and hence constraint (iv) also contributes to the rank of the matrix. Hence its rank was 5.

(d) **He also claimed that not only can he fit the conic, but his metric rectification was also perfect. What do you think might be happening?**

i. A perfect metric rectification is coherent with matrix's rank being 5, since the conic is unique upto a scale, and hence one parameter f can be set to 1 - i.e. there are only 5 independent parameters.

6. Solutions:

(a) **Why do they minimize the Mahalanobis distance?** Criminisi et al. carry out an uncertainty analysis using Mahalanobis distance for robust estimation of parameters **b** and **t.**

i. The Mahalanobis distance differs from Euclidean distance in that it takes into account the correlations of the data set and is scale-invariant.

ii. It is based on correlations between variables by which different patterns can be identified and analyzed. It is a useful way of determining similarity of an unknown sample set to a known one.

iii. Since **t** and **b** are are specified alongwith an uncertanity ellipse, we wish to fit the line through them such that it passes through the vertical vanishing point. At the same time, we need to have a metric which is invariant to scale and arbitrary rotations - hence the use of $Mahalanobis$ distance. A similar analysis for various computations in Q2 can also be done.

7. Solutions:

(a) **Are the two methods equivalent? Comment of how they may be different.**

i. The eight point method registers two planes. One of the registered planes can be used to derive the vanishing line of the plane. Subsequently, the other plane gives the a vanishing point in a direction away from the plane. Thus, registering two planes by the eight point method allows us to (indirectly) specify the two requirements of **[?]**, namely, (i) the vanishing line of a plane and (ii) the vanishing point of a direction away from the plane.

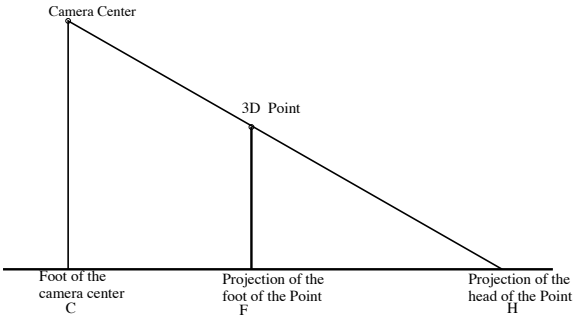

Figure 14: Determining the height

(b) **Does the eight point method in [2] provide any extra information?**

i. The extra degrees of freedom obtained as a result of independent registration of the two planes give us constraints on:

A. translation of the planes along the coordinate axes, and

B. rotation of the coordinate system of one plane with respect to the other

ii. Clearly, these extra constraints are not necessarily required for camera calibration.

(c) **Show that basic methods of probing length ratios along a reference direction used in the two methods are essentially the same.**

i. Once the camera center coordinates have been determined, in either affine or Euclidean terms, the coordinates of 3D points in the scene can be computed using simple geometry. The user clicks a point (call it the head) and its projection on the X-Z plane (call it the foot) in the image along the X Y direction, for example the head and the foot of a person (see Figure 3). Now using the homography $\mathbf{H^{-1}}$ (from the image plane to the X-Z horizontal plane) we can transfer the head and the foot of the point on to the horizontal (X -Z plane). We can then use simple similar triangles to calculate the height (Y coordinate) of the point.

ii. On the other hand, the basic method of probing lengths along a reference direction - Section 2.1: Measurement between parallel planes of [**?**] uses the same concept of similar triangles as discussed in Question 1.iv

CSL840 Major Report

May 8, 2011

Submitted by: Anant Jain, 2008EE10330

Degree of Originality: This is to ascertain that this assignment has been done solitarily by me, and no discussions contributed to this assignment. The papers read beforehand have been cited at appropriate places.

1. **Viola-Jones method of face detection**

   (a) Explain how the Adaboost algorithm has been used for this problem. Why does it work ?

      i. AdaBoost is adaptive in the sense that subsequent classifiers built are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost calls a weak classifier repeatedly in a series of rounds $t = 1, \ldots, T$ from a total T classifiers. For each call a distribution of weights D_t is updated that indicates the importance of examples in the data set for the classification. On each round, the weights of each incorrectly classified example are increased (or alternatively, the weights of each correctly classified example are decreased), so that the new classifier focuses more on those examples. Drawing an analogy between weak classifiers and features, AdaBoost is an effective procedure for searching out a small number of good "features" which nevertheless have significant variety.

      ii. There are 160,000 rectangle features associated with each image sub-window as described in [3], a number far larger than the number of pixels. The hypothesis is that a very small number of these features can be combined to form an effective classifier. In order for the weak learner to be boosted, it is called upon to solve a sequence of learning problems. After the first round of learning, the examples are re-weighted in order to emphasize those which were incorrectly classified by the previous weak classifier. The final strong classifier takes the form of a perceptron, a weighted combination of weak classifiers followed by a threshold. The training error of the strong classifier approaches zero exponentially in the number of rounds.

      iii. The key advantage of AdaBoost as a feature selection mechanism, is the speed of learning. In each round the entire dependence on previously selected features is efficiently and compactly encoded using the example weights.

   (b) What is the role of the attentional cascade?

      i. The key idea of an attentional cascade is that smaller, and therefore more efficient, boosted classifiers can be constructed which

reject many of the negative sub-windows while detecting almost all positive instances. Simpler classifiers are used to reject the majority of sub-windows before more complex classifiers are called upon to achieve low false positive rates. The initial Adaboost threshold is modified so as to have classifiers adjusted to detect 100% of the faces with a false positive rate of 50%. These classifiers are put in a cascade, wherein a negative result from any stage leads to immediate rejection of the sub-window.

ii. The key advantage of an attentional cascade is the dramatic speed-up of the detector by focusing attention on promising regions of the image. More complex processing is reserved only for these promising regions.

(c) Suppose that you have to use this method for detection and tracking of faces in a surveillance video. How would you integrate a tracker with this scheme to ensure temporal consistency?

i. Tracking involves prediction and update for which filters like Kalman filter have been used. Tracking approaches can also be model-based, (for example, using statistical models), or exemplar-based. One tracker disucssed in class was the Kanade-Lucas Tracker, and in general various approaches for tracking features are available. Thus, we can assume that we have a good tracker available to us alongwith the Viola-Jones face detector.

ii. We can use the face detector to initialize the tracker - i.e. as seen above, upon successful determination of a face in a video frame, we automatically get a set of (reliable, good) face features which can serve as the initialization for the tracker. To make the tracker even more robust, the face detector can keep processing frames alongside, and the detection information can be integrated at each time step as the tracker parameters are being propagated, that is, the probabilities are accumulated over time. This would cause the algorithm to continuously detect faces even in frames where the frame-based face detector would fail. The detection information provides knowledge of the appearance of new faces to the temporal framework, which can be readily incorporated whenever they appear by a process of updating.

2. **Graph-cut formulations:** In this question, we arrive at an energy function for both the cases, minimization of which can be done by regular graph cut methods like $\alpha - expansions$ to arrive at a local minimum in the strong sense. Vladimir Kolmogorov and Ramin Zabih give methods to solve the Energy functions described below in [1] and [2].

(a) **Give a graph-cut formulation for 3D reconstruction from two calibrated images.**

i. Given two calibrated images, say L and R, consider a pixel $p \epsilon L$. The epipolar plane through this pixel p containing the epipole e_L of L define a corresponding epipolar line in R, which gives the candidate points q for correspondence in R. Thus, there is a finite set $A = \{< p, q >\}$ of potentially corresponding pixels. The idea is to find a subset of this set A containing only pairs of pixels which correspond to each other. Equivalently, we want to give each assignment $a \epsilon A$ a value f_a which is 1 if the pixels p and q correspond, and otherwise 0. It is clear that having such a subset is enough to have a 3D reconstruction of the scene, provided the cameras are calibrated as given in the question. The appropriate label for the 3D voxel in the set can be obtained by back projecting 3D rays corresponding to these pixels, and finding their intersection.

ii. As done in [1], we will call the assignments in A that have the value 1 active. Let $A(f)$ be the set of active assignments according to the configuration$f$. Let $N_p(f)$ be the set of active assignments in f that involve the pixel p, i.e. $N_p(f) = \{< p, q > \epsilon A(f)$. We will call a configuration f unique if each pixel is involved in at most one active assignment, i.e. $\forall p \epsilon P, |N_p(f)| \leq 1$, where $P = L \bigcup R$, the set of all pixels in two images put together. Note that those pixels for which $|N_p(f)| = 0$ are precisely the occluded pixels.

iii. It is possible to extend the notion of $\alpha - expansions$ to this representation. For an assignment $a =< p, q >$ let $d(a)$ be its disparity: $d(a) = (q_x - p_x, q_y - p_y)$, and let A^α be the set of all assignments in A having disparity α. A configuration f' is said to be within a single $\alpha - expansion$ move of f if $A(f')$ is a subset of $A(f) \bigcup A^\alpha$. In other words, some currently active assignments may be deleted, and some assignments having disparity α may be added. Thus all we are left to do is specify an energy function for the problem.

iv. The energy function for a configuration f can be given by $E(f) = E_{data}(f) + E_{occ}(f) + E_{smooth}(f)$. If $I(p)$ denotes the intensity of pixel p, the three terms are formulated as:

$$E_{data}(f) = \sum_{a \epsilon A(f)} (I(p) - I(q))^2$$

$$E_{occ}(f) = \sum_{p \epsilon P} Cp \cdot T(|Np(f)| = 0)$$

$$E_{smooth}(f) = \sum_{a1, a2 \epsilon N} V_{a1,a2} \cdot T(f(a1) \neq f(a2))$$

The occlusion term imposes a penalty Cp if the pixel p is occluded. The smoothness term imposes a penalty if one assign-

ment is present in the configuration, and another close assignment, having the same disparity, is not. This completes the formulation of the problem for this case.

(b) **Give an object space graph-cut formulation for 3D reconstruction from multiple calibrated images.**

i. The problem of 3D reconstruction from multiple calibrated images was studied in [2]. Suppose we are given n calibrated images of the same scene taken from different viewpoints (or at different moments of time). and let P_i be the set of pixels in the $image_i$, and let $P = P_1 \bigcup P_2 \ldots \bigcup P_n$ be the set of all pixels. It can be noted that each pixel $p \epsilon P$ corresponds to a 3D ray in space. Their problem formulation goal is to find the depth of the point of the first intersection of a 3D ray with an object in the scene for all pixels in all images. Thus, we want to find a labeling $f : P \rightarrow L$ where L is a discrete set of labels corresponding to different depths. Thus, our 3-D reconstruction has pairs of the form $< p, l >$ where $p \epsilon P$ and $l \epsilon L$. It is clear again that having such a configuration f is equivalent to a formulation of the problem where sites are a set of voxels enclosing the object to be reconstructed and the labels are 0 (indicating that either the voxel doesn't belong to the object or is invisible) or 1 (indicating that the voxel is on visible surface boundary of the object)

ii. Again, as discussed in [2], an analogue of set A in part (a) can be a set I consisting of (unordered) pairs of 3D-points $< p_1, l_1 >, < p_2, l_2 >$ "close" to each other in 3D-space. For instance, we can use the constraint that only 3D-points at the same depth can interact, i.e.if $\{< p_1, l_1 >, < p_2, l_2 >\} \epsilon I$ then $l_1 = l_2$.

iii. The energy function for a configuration f can again be given by $E(f) = E_{data}(f) + E_{invis}(f) + E_{smooth}(f)$, where for some constant K,

$$E_{data}(f) = \sum_{<p,f(p)>,<q,f(q)> \epsilon I} min 0, (Intensity(p) - Intensity(q))^2 - K$$

$$E_{invis}(f) = \sum_{<p,f(p)>,<q,f(q)> \epsilon I_{vis}} \infty$$

$$E_{smooth}(f) = \sum_{\{p,q\} \epsilon N} V_{p,q}(f(p), f(q))$$

where, the set I_{vis} satisfies the visibility constraint: if $< p, f(p) >, < q, f(q) > \epsilon I_{vis}$, then $l_1 \neq l_2$. The visibility constraint says that if a 3D-point $\langle p, l \rangle$ is present in a configuration f (i.e. $l = f(p)$) then it "blocks" views from other cameras: if a ray corresponding to a pixel q goes through (or close to)$\langle p, l \rangle$ then its depth is at most l. We will use the set I this part of the

construction of I_{vis}. The set I_{vis} can then be defined as follows: it will contain all pairs of 3D-points $\langle p, l\rangle, \langle q, l\,'\rangle$ such that $\langle p, l\rangle$ and $\langle q, l\rangle$ interact (i.e. they are in I) and $l\,' > l$. Also, the smoothness term involves a notion of neighborhood; we assume that there is a neighborhood system on pixels

$$N \subset \{\{p, q\} | p, q \epsilon P\}$$

This can be the usual 4-neighborhood system: pixels $p = (px, py)$ and $q = (qx, qy)$ are neighbors if they are in the same image and $|px{-}qx| + |py{-}qy| = 1$. This completes the formulation of the problem in multiple calibrated cameras case.

3. **Multi-resolution**

   (a) Use of a pyramid based multi-resolution strategy for image blending (in mosaicing, for example).

      i. An image represented in the usual array of pixel intensities is not suitable for most of the tasks like compression etc. Another natural representation may be by its Fourier transform, with operations applied to the transform coefficients rather than to the original pixel values. But such a representation loses out all the spatial information, and is no good for most of the computer vision tasks. Pyramid based multi-resolution strategy offers, in loose terms, a middle path - it is able to retain spatial localization as well as localization in the spatial—frequency domain by decomposing the image into a set of spatial frequency bandpass component images as described in[6]. Individual samples of a component image represent image pattern information that is appropriately localized, while the bandpassed image as a whole represents information about a particular fineness of detail or scale. The nature of the problem itself demands

      ii. Consider, for example, the apple mosaicing example in [6] discussed in class. The blurred-edge effect in mosiacs is due to a mismatch of low frequencies along the mosaic boundary, while the double-exposure effect is due to a mismatch in high frequencies. In general, there is no choice of transition zone width that can avoid both defects. This dilemma can be resolved if each image is first decomposed into a set of spatial-frequency bands. Then a bandpass mosaic can be constructed in each band by use of a transition zone that is comparable in width to the wavelengths represented in the band. The final mosaic is then obtained by summing the component bandpass mosaics.

   (b) Using muti-resolution feature detectors like SIFT for detection and tracking.

i. In Lowe's paper[5], the idea is to fit a 3-D quadratic through the $scale - space$ in order to determine a feature, since we want features to be invariant to scale, rotation etc. They are well localized in both the spatial and frequency domains, reducing the probability of disruption by occlusion, clutter, or noise. The rationale for considering various scales come from the nature of the images themselves - they contain objects at various scales, and these objects may contain features again at various scales. Moreover, these objects can be at various distances, leading to different sizes. As a result, any analysis procedure that is applied only at a single scale may miss information at other scales. The solution is to carry out analyses at all scales simultaneously.

4. **Robotics competition:** We assume that the goal of the competition is to cover all the landmarks in **least amount of time.**

   (a) **Learning phase:** The following two learning tasks have to be completed before the actual run performance of the robot:

      i. **Step 1: Training the cube detector:** The landmarks given to us are cubes of different sizes and surface textures. We would like to train a classifier similar to[4] as done by Viola-Jones for cube detection in our case. Our primary goal is to be able to learn the concept of a cube by presenting numerous examples of cubes and non-cubes in a setting similar to the discussion in Question 1, but for cubes instead of faces.

      ii. **Step 2: Training the cube recognizer:** The first step on-site would be to train a classifier to recognize one cube from the other. Since it is known that different cubes have different textures, we assume that such textures are visually discernible and uniform for all 6 faces of a given cube. Thus, if step 1 was analogous to face detection problem, this step is analogous to face recognition problem. However, this problem of cube recognition is probably way simpler than of face recognition (using eigenfaces to reduce the dimension of the problem, and looking for nearest neighbour in the eigenface space,) and we can adopt simple time-efficient techniques like matching of SIFT features on the surface of the cube in a nearest neighbour sense. Either way, the design of this step is dictated by the computational power available to us, the latter option being computationally cheaper. Also, we can keep a tab of the size of the cubes upto a ratio, by ensuring that each cube is scanned at a constant distance from the camera.

   (b) **Run phase:** Since we are allowed to carry our own stereo cameras, we can assume that the two camera internal parameters are fixed and known to us, say C_1 and C_2. The algorithmic strategy can be as outlined below:

i. **Step 1: Detecting the next target in the frame:** This step requires us to run the cube detector trained in Step 1 of learning phase. As outlined in Question 1 part (c), the tracker to track the cube can be initialized at the same time, and used for tracking the next target. Also, a similar tracker is initiated for other cubes as and when they appear in the frame while approaching the next target.

ii. **Step 2: Recognizing the next target:** Since we already have an indexed list of the cubes we are supposed to visit, we would like to run the recognizer trained in Step 2 of the learning phase to ensure that we don't end up visitng the same landmark again and again. The idea is to have a list of landmarks to be visited, and use cube recognition to ensure that whenever the next target is identified, we strike it off from the list to avoid any re-visits.

iii. **Step 3: Estimating the position of the next target:** Next, we need to estimate the 3D world position of the cube. This problem has been discussed in class for the case of two stereo calibrated cameras. The idea is to back project two 3D-rays corresponding to the same pixel p common to a 3D-word point in both the images (found by matching SIFT features, for instance), and look for the intersection in a least squares sense.

iv. **Step 4: Moving towards the target cube:** Since the exact R and t from the current position to the cube is now known, the robot's movement control system can be fed with the instruction to move to the target. Simulataneously, step 1 of this phase keeps looking out for other targets in parallel.

(c) **Optimizations:**

i. We would love to reduce the costs, both computationally and hardware wise, and hence would like to switch to a single camera. With a single calibrated camera, we would orient the robot towards the next target by powering the rotation motors of the bot till the cube comes in the centre of the frame, and then proceeding towards the cube. Since we have a tab on the size of the cubes, we would know the distance to the cube from single view geometry as well (presence of knowledge of one dimension).

ii. For the case of stereo cameras, in case of multiple available targets, we would like to choose the one which is closer to the robot. (greedy solution to a TSP problem.)

References

[1] Vladimir Kolmogorov and Ramin Zabih. Visual correspondence with occlusions using graph cuts. In International Conference on Computer Vision, pages 508–515, 2001.

[2] Vladimir Kolmogorov and Ramin Zabih. Multi-camera Scene Reconstruction via Graph Cuts. In European Conference on Computer Vision, pages 82-96, 2002.

[3] P. Viola and M. J. Jones, "Robust Real-time Face Detection", In International Journal of Computer Vision 57(2), pages 137–154, 2004.

[4] P. Viola and M. J. Jones, "Robust Real-time Object Detection", In Proc. of IEEE Workshop on Statistical and. Theories of Computer Vision, 2001.

[5] Lowe, D. G., "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, 60, 2, pp. 91-110, 2004

[6] E. H. Adelson et al, "Pyramid methods in image processing"