Fnu Tulha
CAP 6419 Assignment 2

Introduction and Implementation

For this assignment, we were instructed to implement the algorithms for affine and metric rectification. In this report, I will explain the algorithms implemented with a strong focus on the discussion of experimental results.

Affine Rectification

In the algorithm for this rectification, we were provided with a projective image from which our goal was to remove the projective distortion up to an affinity. The key idea was to identify the two pair of lines that were parallel in the world plane but not in the image plane. Once these lines had been identified, I attempted to locate the points at infinity in the image plane. Since parallel lines intersect in projective space and the image had been stored projectively, the intersection of the aforementioned lines would give us the ideal points. The line through these ideal points would end up giving us the equation of the line at infinity.

Once we obtained the equation of the line at infinity, we could map it to its canonical position using a transformation. This transformation would be denoted by

$$H_{affine} = \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{matrix}$$

The steps of the algorithm are outlined below:

1. Select 4 points in the given image to identify two sets of parallel lines. L_1 through L_4.
2. Given that L_1 is parallel to L_2 and L_3 is parallel to L_4, we can evaluate the ideal points using the intersection of the parallel lines.
3. The vector product (cross) of L_1, L_2 and L_3, L_4 will provide us with the intersection (ideal) points, m_1 and m_2.

Since points are dual to lines and vice versa in projective space. $l_{inf} = m_1 \; x \; m_2$ will give us the equation of the line at infinity. After obtaining the equation to this line, we will set up the H matrix as mentioned earlier and will apply it to the image plane for every pixel. In MATLAB, the implementation of the function '**imtransform'** performs this very operation.

$$I_{affineRectified} = \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{matrix} * I_{projective}$$

An important thing to mention here is that this function has a different format for a transformation and requires a transposed transformation to be able run correctly.

Metric Rectification

This algorithm was divided up into two parts 1a, 1b and 2. Since the images obtained from the previous algorithm are projectively undistorted, we will use them in our algorithm for the first part.

Metric Rectification Version 1a:

In this version, we use an image that has been rectified up to an affinity as in part I of this assignment. We understand that the dual conic encodes all the information necessary to recover an image up to a similarity.

$$\begin{aligned}
\mathbf{C}_\infty^{*}{}' &= \left(\mathbf{H}_P\mathbf{H}_A\mathbf{H}_S\right)\mathbf{C}_\infty^{*}\left(\mathbf{H}_P\mathbf{H}_A\mathbf{H}_S\right)^{\mathsf{T}} \\
&= \left(\mathbf{H}_P\mathbf{H}_A\right)\mathbf{H}_S\mathbf{C}_\infty^{*}\mathbf{H}_S^{\mathsf{T}}\left(\mathbf{H}_P\mathbf{H}_A\right)^{\mathsf{T}} \\
&= \left(\mathbf{H}_P\mathbf{H}_A\right)\mathbf{C}_\infty^{*}\left(\mathbf{H}_P\mathbf{H}_A\right)^{\mathsf{T}} \\
&= \begin{bmatrix} \mathbf{K}\mathbf{K}^{\mathsf{T}} & \mathbf{K}^{\mathsf{T}}\mathrm{v} \\ \mathrm{v}^{\mathsf{T}}\mathbf{K} & \mathrm{v}^{\mathsf{T}}\mathrm{v} \end{bmatrix}
\end{aligned}$$

Since our dual conic remains the same under a similarity transform we end up with the above equation. However, we should note that we have removed projective distortion in part I of this assignment which means that we only need to be concerned with H_a. We also notice that this leads to v = 0 and the following result is derived.

$$\begin{pmatrix} l_1' & l_2' & l_3' \end{pmatrix} \begin{bmatrix} \mathbf{K}\mathbf{K}^{\mathsf{T}} & 0 \\ 0 & 0 \end{bmatrix} \begin{pmatrix} m_1' \\ m_2' \\ m_3' \end{pmatrix} = 0$$

$$\left(l_1'm_1', l_1'm_2' + l_2'm_1', l_2'm_2'\right)\left(k_{11}^2 + k_{12}^2, k_{11}k_{12}, k_{22}^2\right)^{\mathsf{T}} = 0$$

This equation holds for two pair of lines that are orthogonal in the image plane. We select these pairs of lines by selecting 3 points, the vector product of which results in a pair of lines that satisfy the above relationship. It is important to mention that we are attempting to solve for the matrix S = K.K' where S is going to be the null vector to the above system of equations. Since S is symmetric we can simply it to be equal to s = (s11, s12, s13). Following from the above equation we obtain a 3 by 4 matrix that needs two sets of orthogonal lines to solve. A selection of 6 points in the MATLAB code would lead to 2 pairs of lines as shown above.

After obtaining S we wanted a decomposition such that K*K' = S. According to the textbook, the Cholesky decomposition, popularly known as **'chol'** in MATLAB achieves this goal. However, due to the matrix S sometimes not satisfying the condition of positive semi definiteness would fail to work. Another function found on the Mathworks website took the matrix S as input and returned the closest positive semi definite matrix as output. However, this method proved to fail for majority of my test cases due to which I used another workaround to solve the problem. By simply taking the product of S*S' we ended up with a symmetric matrix which satisfied the condition of positive semi definiteness. The problems with this approach will be discussed in the reflection section but the results obtained by this workaround were almost satisfactory.

After the matrix K was obtained, the transformation was applied to the image to rectify it.

Metric Rectification Version 1b:

In this version of Metric Rectification, the goal was to use a conic(circle) present in the image plane, take its intersection with the line at infinity and obtain the circular points which would then give me the dual conic. This method proved to be the least efficient and this will be discussed in the reflection section as well.

The method is outlines below:

1. Select five points on the conic present in the picture and use conicfit to obtain C.
2. Select 4 points in an anti clockwise fashion to obtain two sets of lines that are parallel in the image plane (as in affine rectification).
3. Using the vector product, determine the coordinates of the ideal points.
4. Use the modified function from assignment 1 to find the intersection point of the conic and the line at infinity
5. Two imaginary points should be returned which are the circular points. Use C = IJ' + JI' to obtain the equation of the dual conic.
6. Using singular value decomposition, find the transformation U such that C*_inf = U*S*U'.
7. Perform an arithmetic trick to modify U and U' such that S becomes the identity matrix
8. Apply the transformation to the image to remove the affine distortion

Metric Rectification Version 2:

This method was slightly different from the first two versions, 1a and 1b since it performed the entire rectification pipeline of a projective image in one step. The biggest challenge here was to attempt to use 5 sets of orthogonal lines to satisfy the constraint of the system of equations.

$$\begin{pmatrix} l_1' & l_2' & l_3' \end{pmatrix} \begin{bmatrix} \mathbf{K}\mathbf{K}^\mathsf{T} & \mathbf{K}^\mathsf{T}\mathrm{v} \\ \mathrm{v}^\mathsf{T}\mathbf{K} & \mathrm{v}^\mathsf{T}\mathrm{v} \end{bmatrix} \begin{pmatrix} m_1' \\ m_2' \\ m_3' \end{pmatrix} = 0$$

We came across the above equation earlier but with v = 0 since there was no projective distortion in the image. Since this pipeline takes a projective image as input, the value of v is going to be unknown. The following equation results in a 1 by 6 matrix with c giving the coefficients of the conic. We would need 5 sets of equations here and each constraint would involve the equations of two lines hence 2*5 = 10 lines will be present. Since two orthogonal lines can be defined using a minimum of 3 points we will need 15 such points.

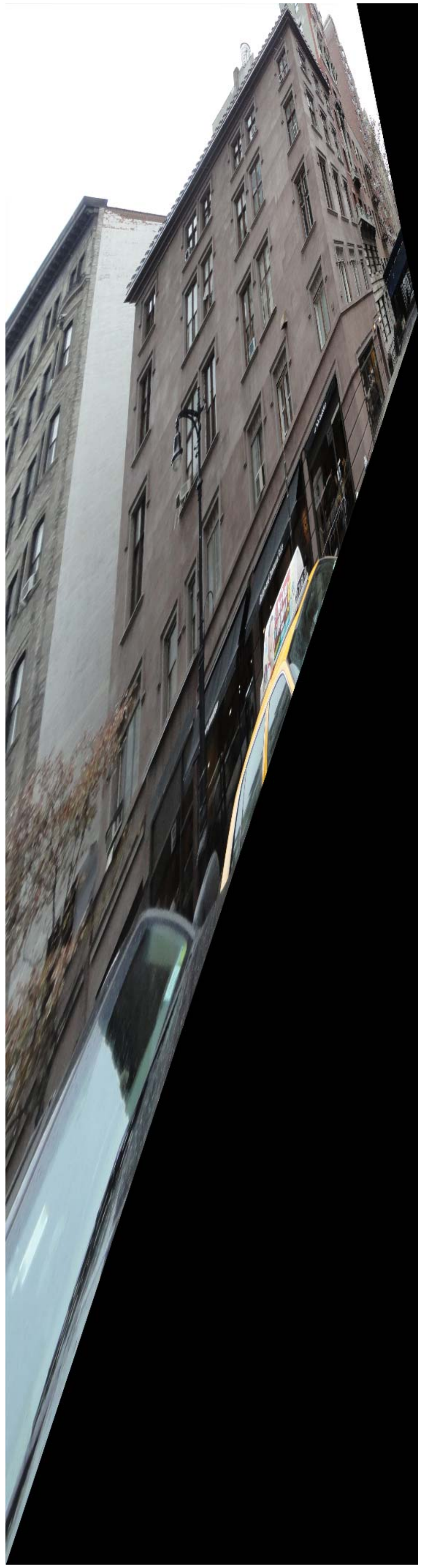
$$\left(l_1'm_1', 0.5\left(l_1'm_2' + l_2'm_1'\right), l_2'm_2', 0.5\left(l_1'm_3' + l_3'm_1'\right), 0.5\left(l_2'm_3' + l_3'm_2'\right), l_3'm_3'\right)\mathrm{c} = 0$$

After solving this system of equations by finding the null vector of c, we will use singular value decomposition on the conic at infinity. The transformation that results will rectify the image by removing both projective and affine distortions.

Results

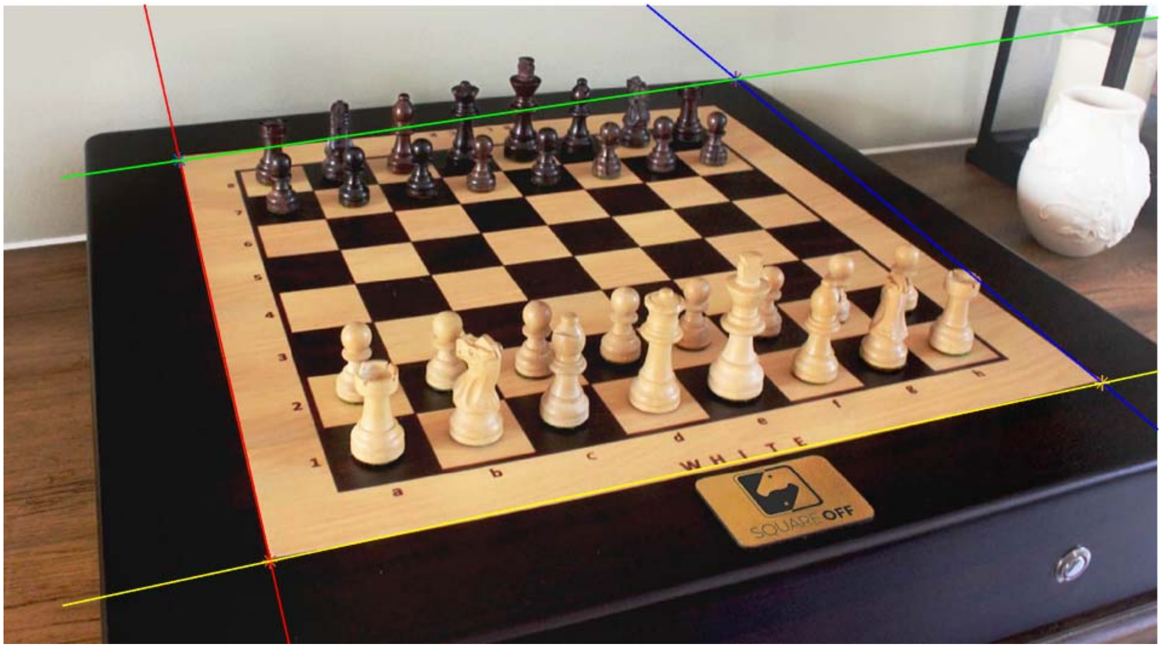
The result will be displayed below in the following format: ***Image with selected points, Image with plotted lines and the rectified image***

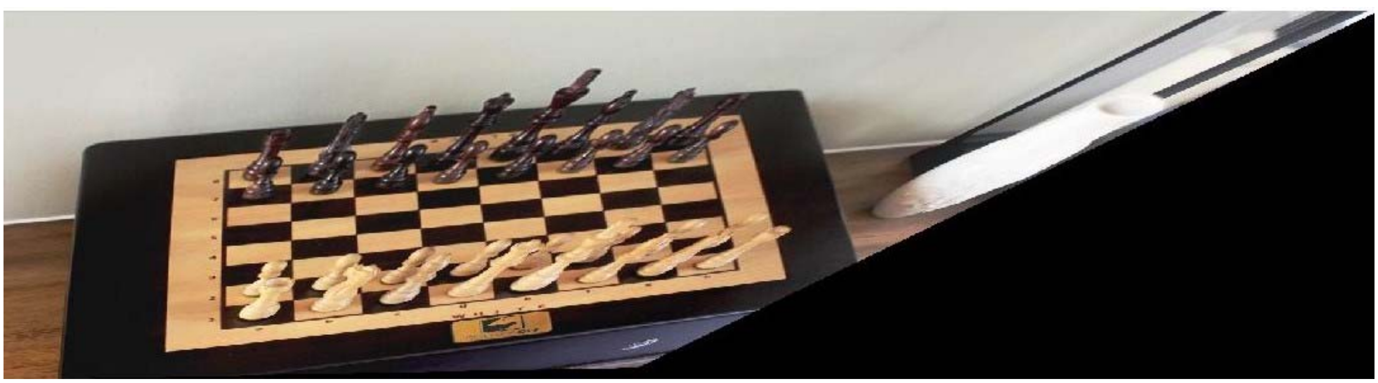
- **Affine Rectification**
  - **Building 1**



- **Building 2**

- **Chessboard**

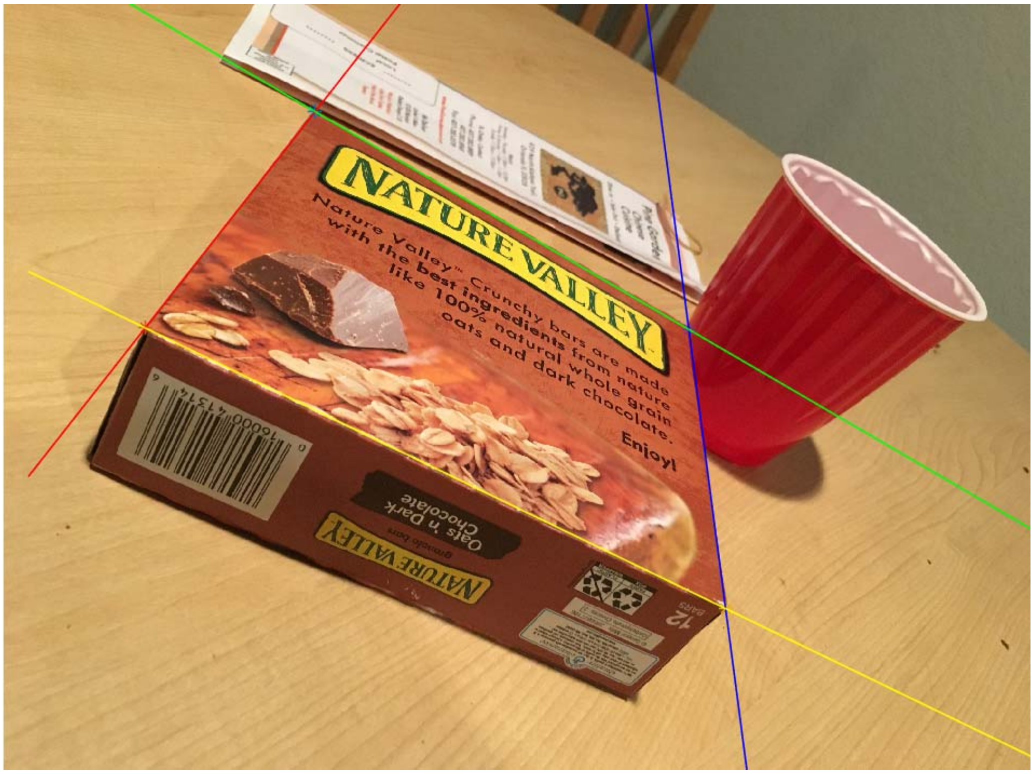

WHITE
SQUAREOFF



- **Crop circles**

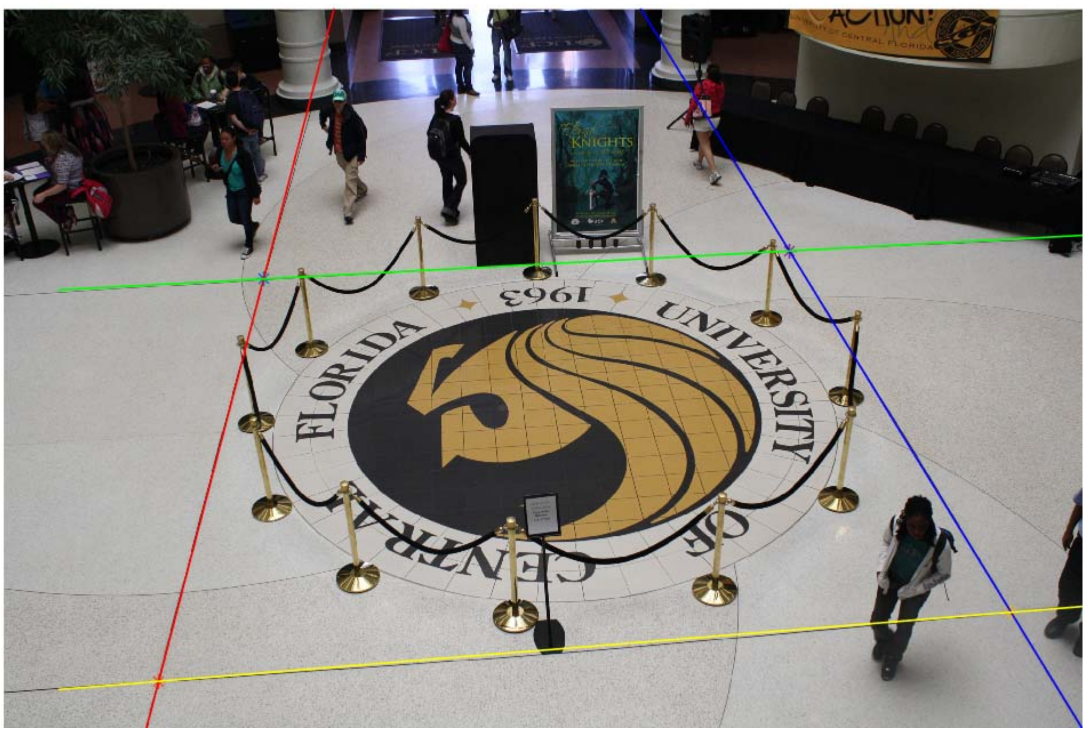
- **Home**

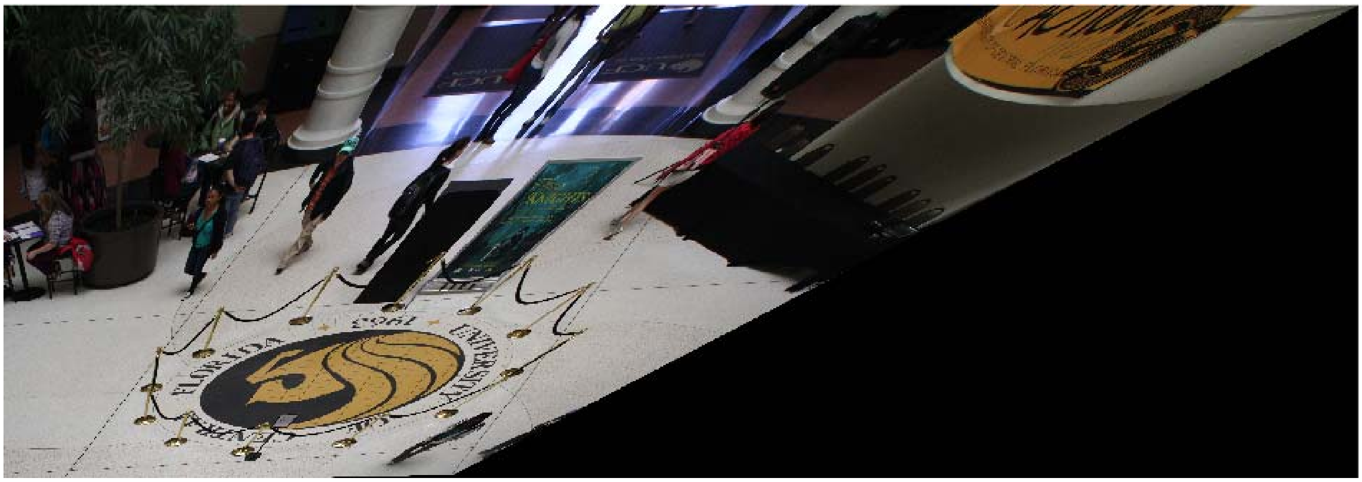

NATURE VALLEY
Nature Valley™ Crunchy bars are made with the best ingredients from nature like 100% natural whole grain oats and dark chocolate.
Enjoy!
Oats 'n Dark Chocolate
NATURE VALLEY
12


NATURE VALLEY
Nature Valley™ Crunchy bars are made with the best ingredients from nature like 100% natural whole grain oats and dark chocolate.
Enjoy!
NATURE VALLEY
Oats 'n Dark Chocolate
12

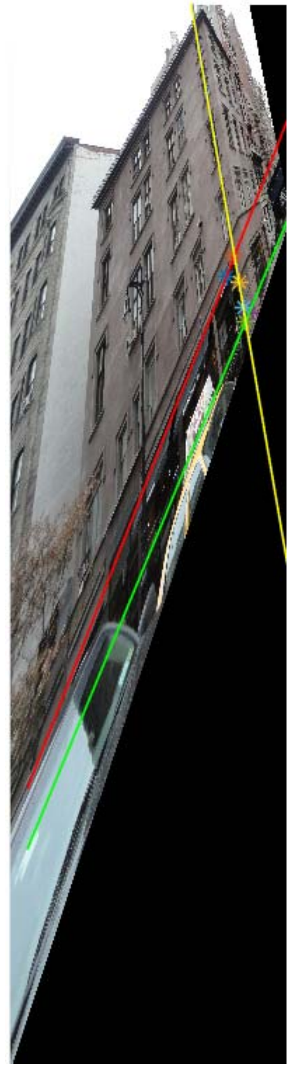
- **UCF SU**

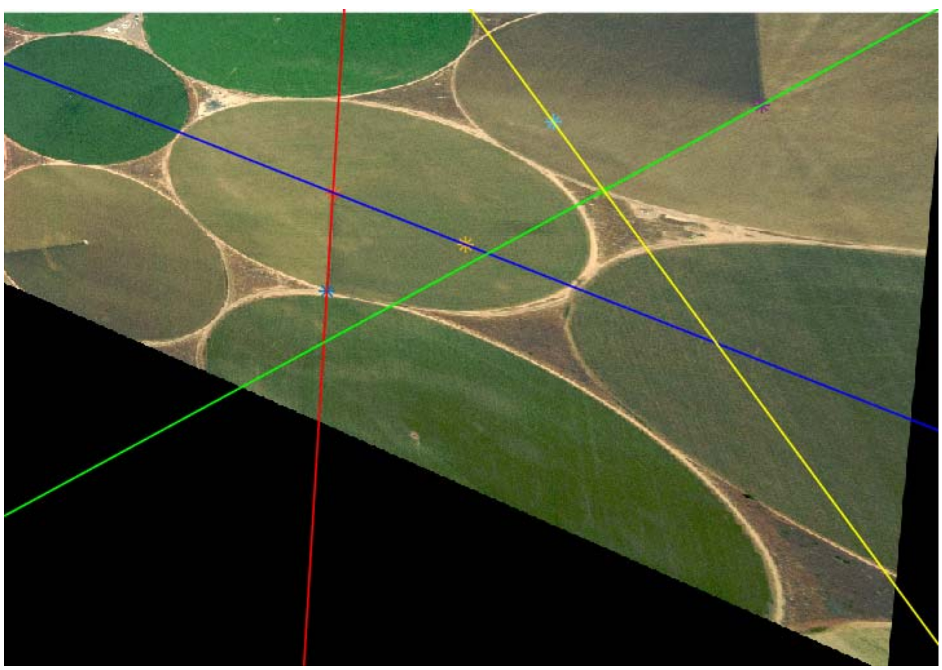

FLORIDA
UNIVERSITY
OF
CENTRAL
1963
KNIGHTS

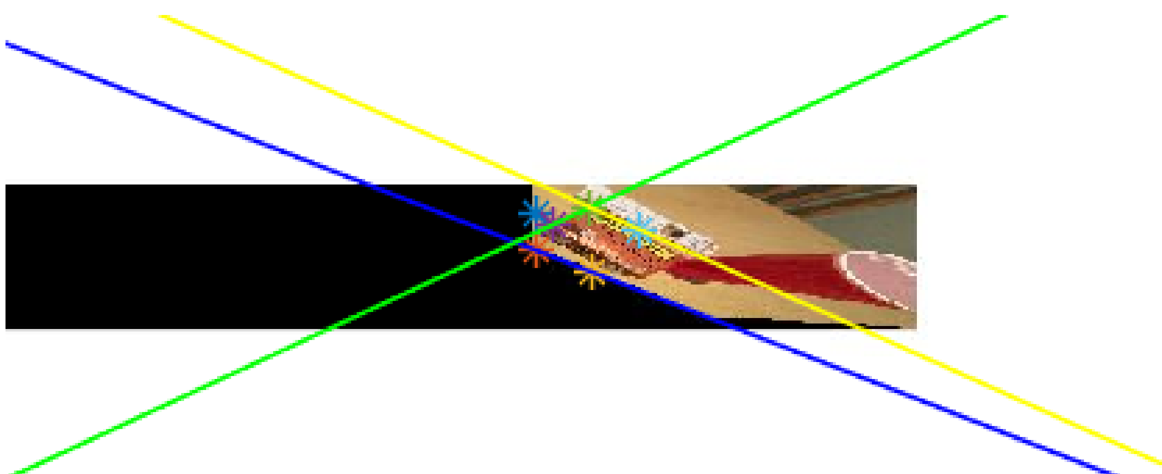

FLORIDA
UNIVERSITY
OF
CENTRAL
1963
KNIGHTS

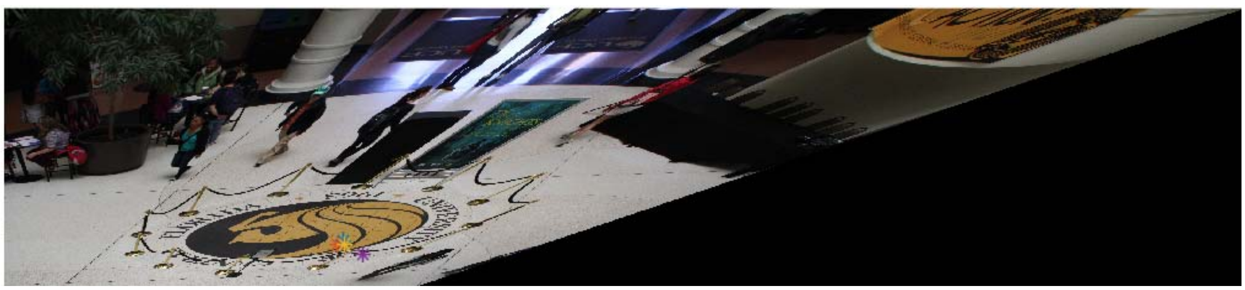


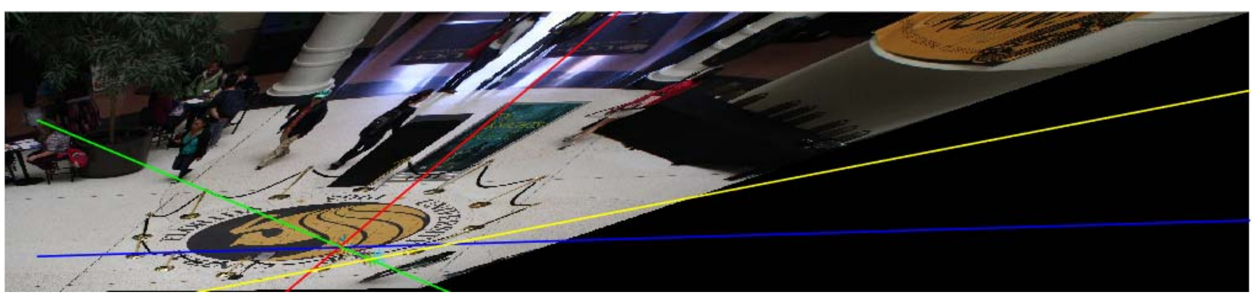
- **Metric Rectification 1a**
  - **Building 1**



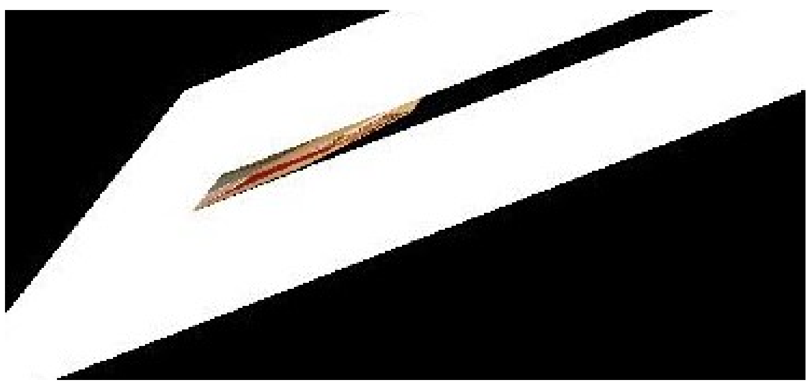


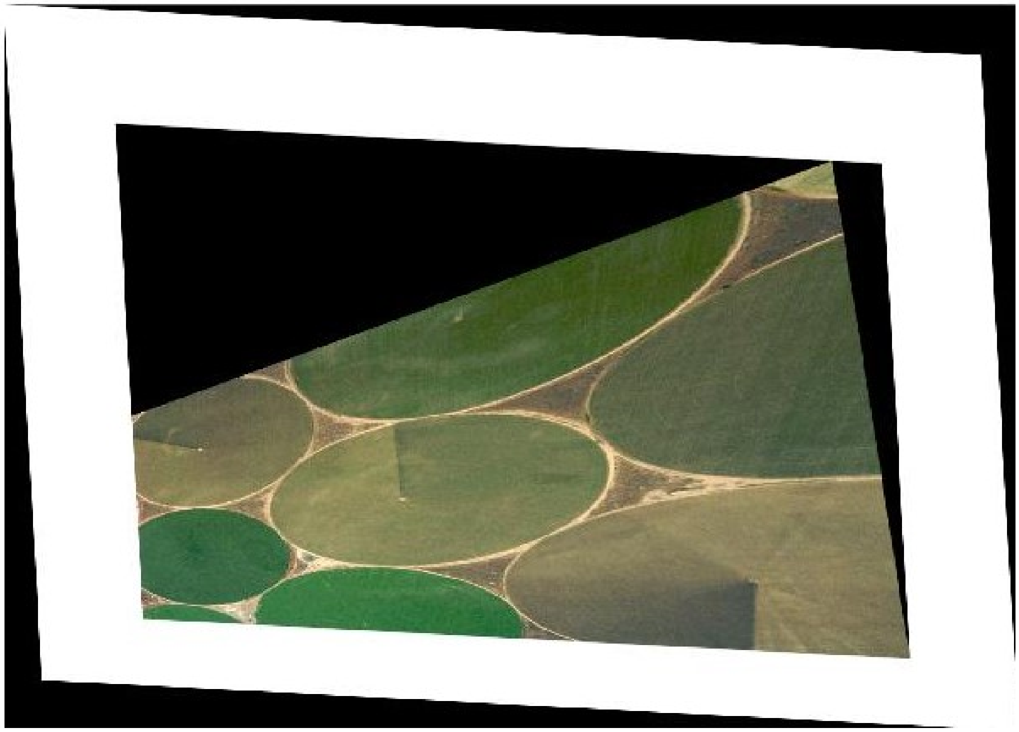






- **Metric Rectification 1b**





- **Metric Rectification 2**

Selected points for pairs of orthogonal lines

Rectified Image

Selected points for pairs of orthogonal lines

Rectified Image

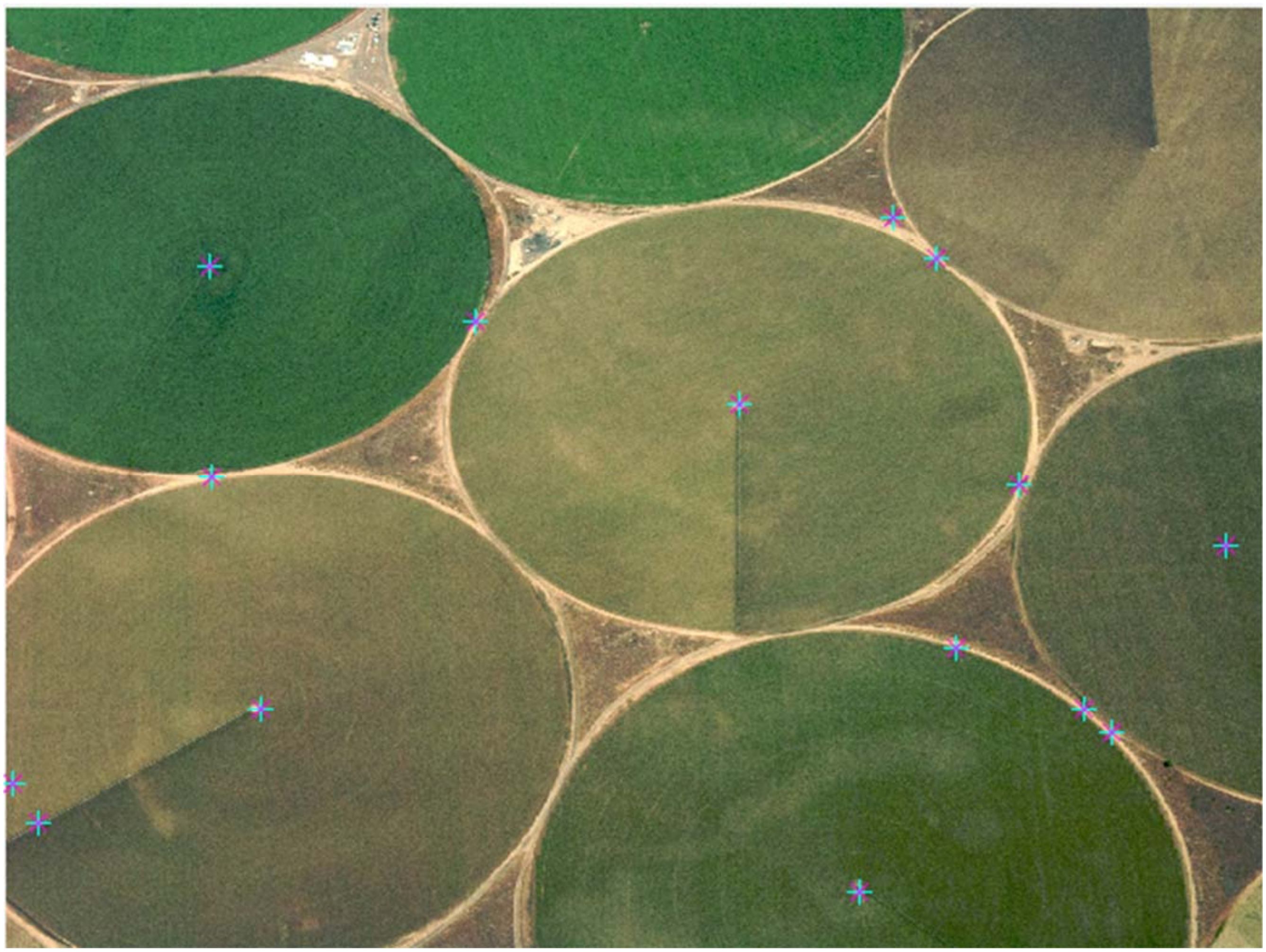
Selected points for pairs of orthogonal lines

Rectified Image

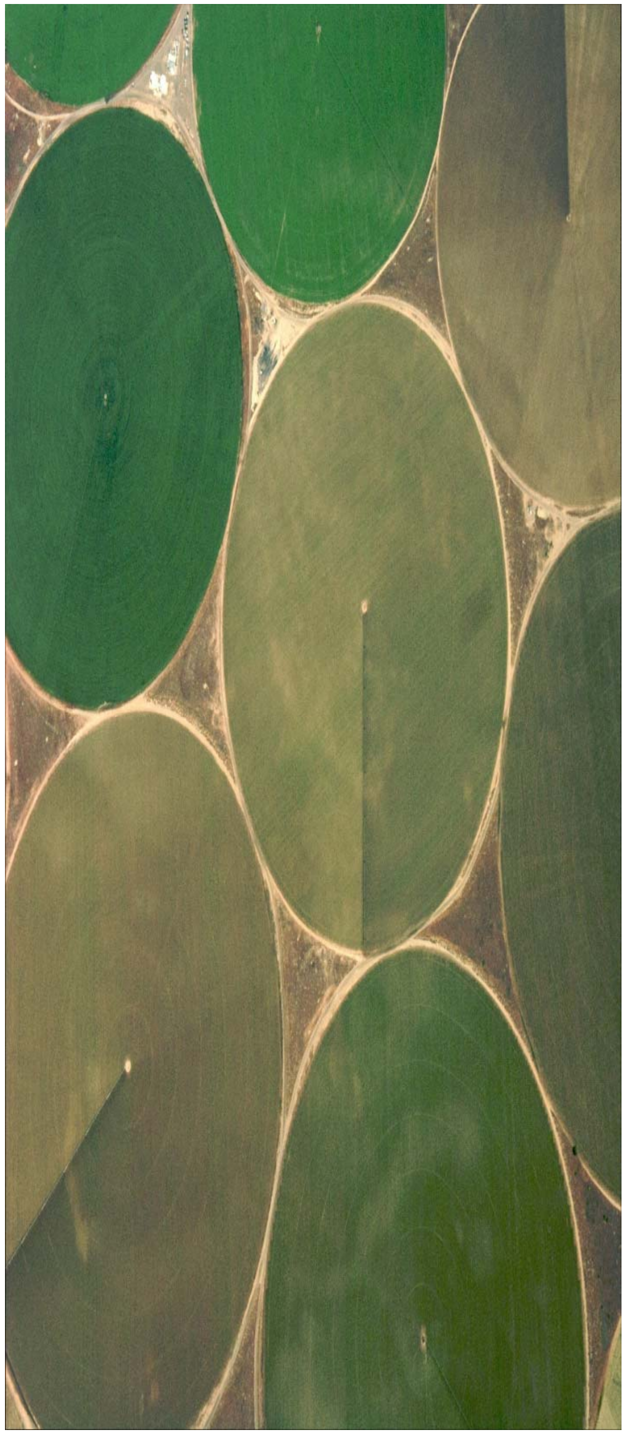
Selected points for pairs of orthogonal lines

Rectified Image

Fnu Tulha



Selected points for pairs of orthogonal lines



Rectified Image

Selected points for pairs of orthogonal lines

Rectified Image

Selected points for pairs of orthogonal lines

Rectified Image

Note: If the images are not clear please check the results folder as all the images should be there.

Special Observations/Reflection

There were many observations one could make from the experiments. They have been listed below:

- For affine rectification whenever we are selecting a set of points to find the line at infinity and there are multiple planes in the picture as in the picture of the building i.e the ground plane and wall plane are two separate planes both incorporated in the same image. In this case, we cannot apply the same transformation to both planes and expect to get the same results because the lines that are forming are emerging out of the 2D plane at an angle. One can visualize this problem by imagining two orthogonal planes and a line on both these planes joining up.
- For metric rectification 1a, orthogonal lines had to be selected that were linearly independent. For instance, in most of the images such as the chessboard, if one selects two orthogonal lines on the square marks then the subsequent selection of the pair of lines should not be from the boxes since if both lines are linearly dependent the solution to the system L.s = 0 would not be valid.
- For rectification 1b, the circular points had to be determined which was very problematic since calculation of the ideal points was subject to point selection errors and even the tiniest deviation would result in a different value for the I and J circular points and subsequently the dual conic matrix.
- The Cholesky decomposition in metric rectification 1a required a positive semi definite matrix as mentioned earlier. There was no systematic method to convert a matrix to this form. However, symmetric matrices 'seemed' to have the property of being positive semi definite as well. Therefore, to decompose C, one could simply decompose the product of C and C' to get around the above-mentioned error. It is important to mention that the errors noticed in rectification was exactly because of this reason. The accumulation of error due to an extra factor of C' would result in a different K matrix and subsequently a different transformation and an incorrect rectification.
- Rectification 2 was the most successful out of the three approaches but it is obvious that this is most likely the most error prone approach. Choosing 5 sets of orthogonal lines by selecting a minimum of 15 points is neither practical nor recommended. Just a tiny change in the values of the selected points would result in large variations in the rectification transform.
- Some interesting observations:

Affine Rectification at 0.5x the resolution.

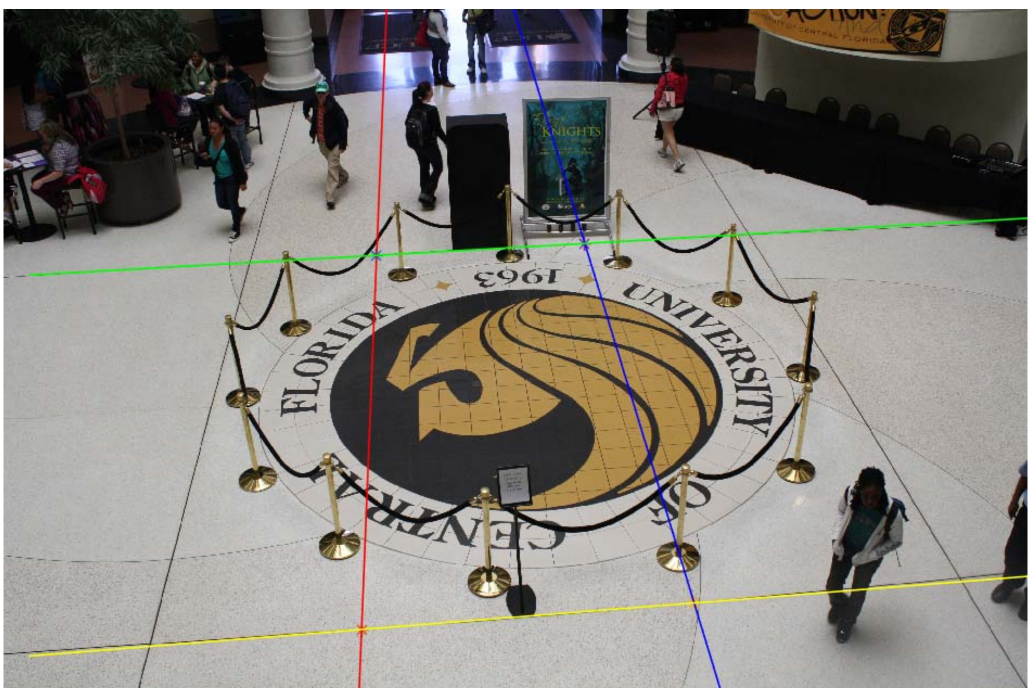
Affine Rectification at 0.1x the resolution.

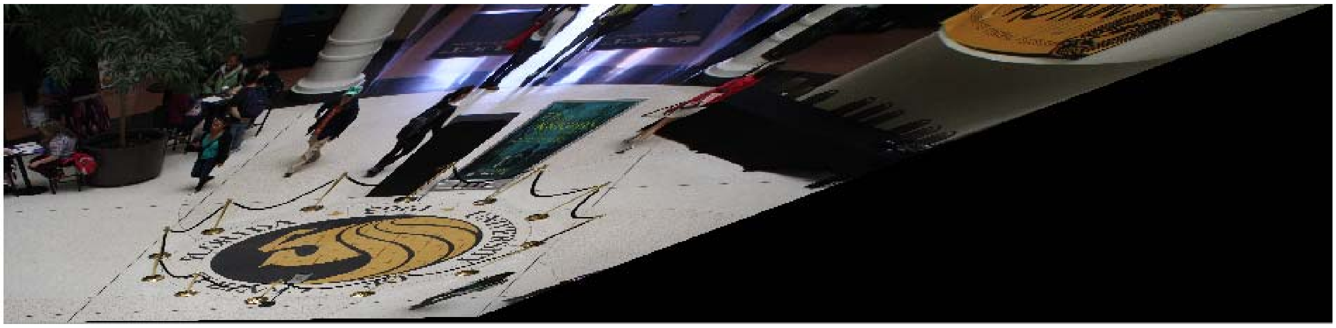
When identical points were chosen for two images with different scaled down resolutions. The image that was scaled down by 0.1x the resolution showed an inaccurate rectification compared to the image that was scaled down to 0.5x.

Very distant parallel lines were chosen for affine rectification.

As a result the affine rectification that resulted was very inaccurate.

Very close parallel lines were chosen for affine rectification.





The affine rectification that resulted was more accurate than its counterpart showing how lines that are selected using points that are far apart will be relatively erroneous.

References

- Multi View Geometry in Computer Vision, Richard Hartley and Andrew Zisserman
- The positive semi definiteness of a matrix https://math.stackexchange.com/questions/168717/about-positive-semidefiniteness-of-one-matrix?rq=1
- Positive Semi-definite Matrix Problem https://www.mathworks.com/matlabcentral/answers/84287-positive-semi-definite-matrix-problem
- Nearest semi definite positive matrix implementation https://www.mathworks.com/matlabcentral/fileexchange/42885-nearestspd
- Test image Building 2 http://www.techniserve.co.uk/projects/fixed-wire-Test-and-rectification-work
- How to apply a 2D -2D homography matrix to an image? https://dsp.stackexchange.com/questions/21703/how-to-apply-a-2d-2d-homography-matrix-to-an-image
- https://www.mathworks.com/matlabcentral/answers/291814-error-using-zeros-maximum-variable-size-allowed-by-the-program-is-exceeded?s_tid=answers_rc1-1_p1_Topic
- Ginput2 function MATLAB http://de.mathworks.com/matlabcentral/fileexchange/20645-ginput2-m-v3-1--nov-2009-