School of Instrumentation

Devi Ahilya Vishwa Vidyalaya, Indore

COMPUTATIONALLY EFFICIENT VISUAL METROLOGY USING SINGLE VIEW IMAGE

An Interim project

By

Arpit Mourya (21IOT6003)

Yashdeep Kumrawat (21IOT6031)

Under the Guidance of

Mr. Pranav Kumar Pandey, Scientist ‘E’

HEMRL Pune

Approved by

Mrs. Sandeep Kaur, Scientist ‘F’

Group Director, HEMRL

Acknowledgements

First of all, We would like to thank Dr. A P Dash, OS & Director, HEMRL and Dr. Gajesh Joshi, Scientist ‘F’, HRD HEMRL for giving us the opportunity to work with HEMRL.

We are grateful to Mrs. Sandeep Kaur, Scientist ‘F’, Group Head Instrumentation for her constant support during the entire project duration.

We would like to express our appreciation to Mr. Pranav Kumar Pandey, Scientist ‘E’ for his valuable support, advice and guidance during this project work.

We would like to express our sincere gratitude to our supervisors, Mr. Vivek Nagdive, Scientist ‘C’ and Mr. Ravi Ranjan, Scientist ‘C’. Their unwavering enthusiasm, guidance, and dedication to finding the best solutions to the challenges we faced were instrumental in achieving the results presented in this report. We also appreciate their friendly support throughout our internship.

We would like to thank our Head of Department Dr. Ratnesh Gupta for his constant motivation throughout our academic journey.

We would also like to give thanks to Mr. Vipul Kumar (Senior Technical Asst.) and Mr. Rajveer Meena (Technician) for contributing, in measurements and validations for method presented in this report.

Our special appreciation goes to our family and friends. They have been a consistent source of love, care, support and affection, a person could ever ask for.

We would also like to thank field staff of the Instrumentation division. They have helped us in giving practical insight to this project. We thank all those who are directly or indirectly involved in shaping up this project.

Abstract

The work described here develops the method of computing world measurements from photographs of scenes. The method developed here addresses a number of different situations ranging from where no scene metric information is known, to cases where some distances are known but there is not sufficient information for a complete camera calibration.

Single view metrology investigates how planar measurements can be taken from uncalibrated single camera images. The measurement method developed is independent of the camera's internal parameters: focal length, aspect ratio, principal point and skewness.

Object tracking is the area of interest when working with videos. HSV (hue, saturation value) based color segmentation is used in this algorithm which is very versatile method for detecting objects, especially situations where the object is not well known to be trained for object detection.

We assume that the vanishing line of a reference plane in the scene may be determined from the image, together with a vanishing point for a reference direction. The work has a variety of applications like: forensic measurements, measurements at high human risk area, virtual modeling and architectural measurements.

Objective

This research investigates a single-view metrology method for object dimension measurement using HSV color space segmentation. This approach aims to overcome limitations of traditional methods requiring perpendicular camera placement. The objective is to develop a system that can accurately track and measure objects regardless of camera orientation relative to the object. By leveraging HSV color space segmentation, the system will isolate objects based on their color properties, enabling dimension extraction from a single image captured at any angle.

- Implement method on videos: Apply the HSV segmentation algorithm to video data for object tracking and dimension measurement across various viewpoints.
- Validate with existing data: Compare the system's measurements with known dimensions from existing data sources to confirm accuracy.
- Evaluate accuracy across angles: Assess the impact of camera angles on measurement accuracy by capturing videos from different orientations and comparing results.
- Quantify and analyze errors: Calculate errors between measured and actual dimensions, identify error sources, and explore methods for improvement.

Contents

Chapter 1

Introduction

1.1 Single View Metrology

This chapter describes how aspects of the affine 3D geometry of a scene may be measured from a single perspective image.

The technique described here concentrate on scenes containing planes and parallel lines, although the methods are not so restricted. Measurement of object present in a frame can be calculated using known reference object present in the frame. Various computer vision applications require the measurement of objects perspective image.

Single view metrology refers to the monocular view measurement estimation of objects. Since we have monocular view of the object we have ample information regarding the parameters which are needed to estimate the dimension such as width, height of an object and distance between objects in the image.

1.2 Pinhole camera

A pinhole camera (see fig 1.1) is a basic camera that produces an image through a tiny hole rather than a lens. The hole, also known as the pinhole, is commonly drilled through a thin piece of metal, such as an aluminum or brass sheet. The light going through the pinhole creates an inverted image on a screen or film placed behind it.

Pinhole cameras have various advantages over conventional cameras with lenses. They are small, light, and affordable to produce. They also feature a broad depth of field, which means that objects at different distances from the camera can be in focus simultaneously. Furthermore, pinhole cameras do not suffer from optical distortions, which can degrade image quality in regular cameras.

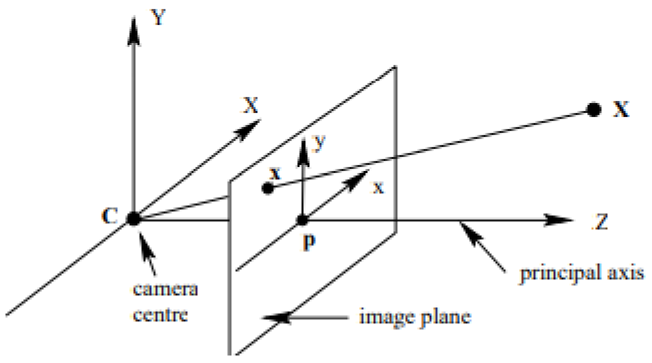


Fig 1.1 Pinhole camera model

However, pinhole cameras do have significant limitations. They have a slower shutter speed than standard cameras, which can cause motion blur in moving objects. The tiny aperture and associated diffraction of light can further reduce image quality.

1.3 Vanishing point

Vanishing lines are formed when at least two parallel lines in real world are imaged using a camera and they appear to intersect each other at a finite point called *vanishing point (see fig 1.2)*. Thus a vanishing point depends only on the direction of a line, not on its position. The line passing through all vanishing points is called as *horizon line*.

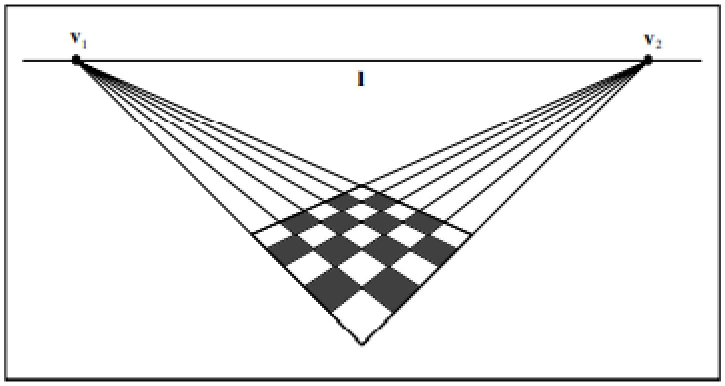


Fig 1.2 Vanishing Points

1.4 Image segmentation

Image segmentation is a fundamental problem in computer vision that includes splitting an image into individual portions or objects. It is an important stage in image analysis since it allows objects to be separated from the background and from one another, as well as facilitates tasks like object identification, tracking, and recognition.

What is image segmentation?

Image segmentation is the process of labelling or classifying each pixel in an image to indicate which object or region it belongs to. This enables the identification of certain objects, textures, or regions within the image.

1.4.1. Semantic Segmentation

Semantic segmentation involves assigning a class label to each pixel in an image, indicating the object or region it belongs to. This type of segmentation is useful for tasks like object detection, scene understanding, and image retrieval.

1.4.2. Instance Segmentation

Instance segmentation involves identifying and separating individual objects within an image, even if they belong to the same class. This type of segmentation is useful for tasks like object tracking, 3D reconstruction, and robotics.

Chapter 2

Methodology

2.1 Assumptions

This section outlines the assumptions made in the methods presented in the document. These assumptions are important to understand the limitations and applicability of the techniques.

- It is assumed that the vanishing line of a reference plane in the scene may be determined from the image.
- There should be at least one object (reference object) with the known dimension lying parallel to the main object.
- The object which is to measured is not point-size.
- Point of main object on ground plane is known.

2.2 Flow chart

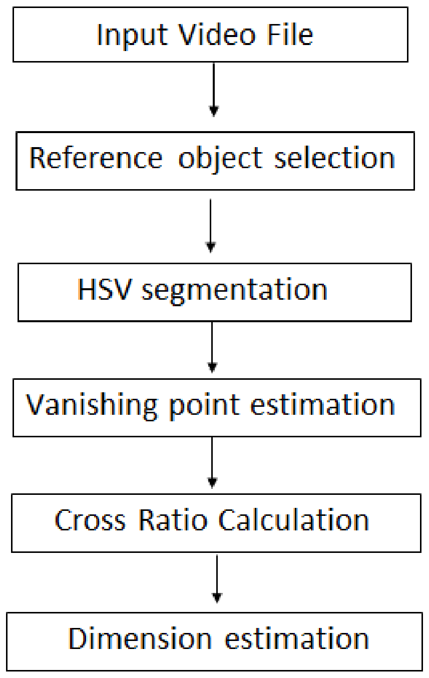


Fig 2.1 Flow chart of method

2.3 HSV Segmentation

The HSV (Hue, Saturation, Value) color space is a cylindrical coordinate system that depicts colors more intuitively and perceptually than the classic RGB (Red, Green, Blue) color space. The HSV color space is especially beneficial for color segmentation and object recognition tasks. HSV Color Space Components: The HSV color space has three components.

Hue (H): The color tone or hue is represented by a number ranging from 0 to 360. Hue is a circular parameter: 0° corresponds to red, 120° to green, and 240° to blue. Hue-Based Segmentation: Separate objects according to their hue values. This is useful for distinguishing between things with distinct color tones, such as red, green, and blue.

Saturation (S): The purity or strength of a color, which ranges from 0 (grayscale) to 1 (completely saturated). Saturation-Based Segmentation: Separate objects based on their saturation levels. This is useful for distinguishing between things with varying degrees of color purity, such as pastel and bright colors

Value (V): Indicates the brightness or lightness of the color, ranging from 0 (black) to 1 (white). Value-Based Segmentation: Organize things according to their value values. This is useful for differentiating things with varying brightness levels, such as light and dark regions.

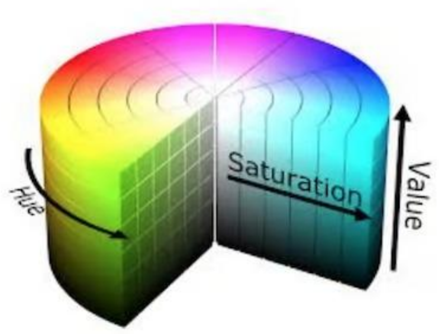
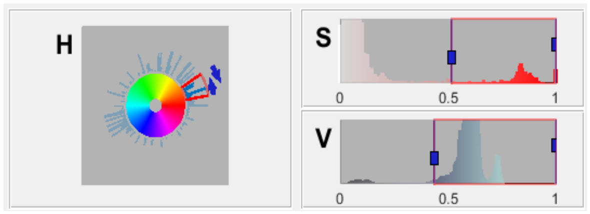


Fig 2.2 HSV Color Space

HSV Color Space Segmentation. HSV color space segmentation (see fig 2.2) separates objects or regions of interest from the backdrop depending on their color qualities. The segmentation procedure typically includes the following steps: Color Space Conversions: Convert the RGB image into HSV color space.

A. Segmented Image

B. Real Image



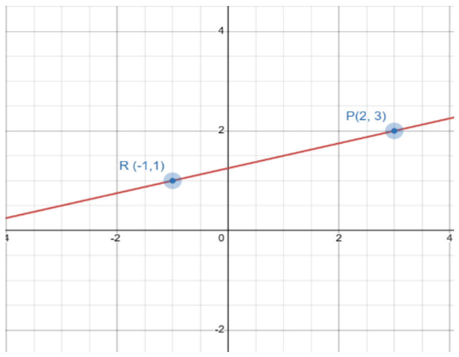
C. HSV selection panel

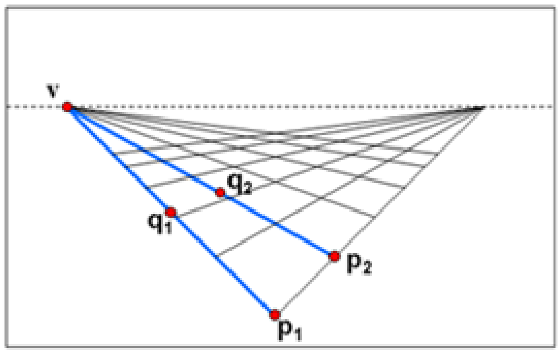
Fig 2.3 Image Segmentation using HSV Color space

HSV Color Space Segmentation Techniques: Several ways can be used to segment the HSV color space, including: Thresholding (see fig 2.3 A).

2.4 Vanishing point calculation

Section 1.3 describes the vanishing point as intersection point of two lines in camera image. To find a vanishing point at least two lines are required.





A. Line passing through two points B. Vanishing points from four points

Fig 2.4 Vanishing point calculation

Lines passing through two points can be calculated from formula

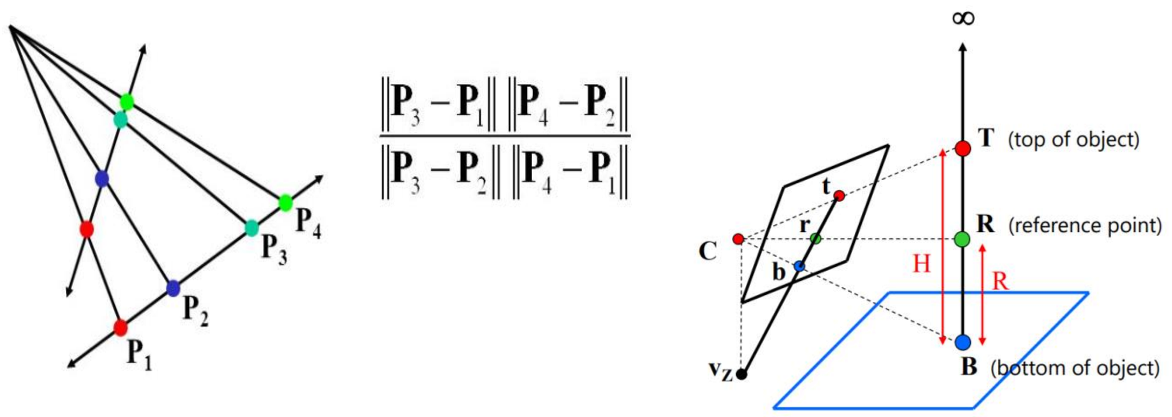
$$L = R\times P = [-1; 1; 1] \times [2; 3; 1]$$

Similarly vanishing point from the intersection of the lines can be calculated.

$$v = (p1 \times q1) \times (p2 \times q2)$$

2.5 Cross ratio calculation

In geometry, the cross-ratio, also called the double ratio (see fig 2.5), is a number associated with a list of four collinear points, particularly points on a projective line. The image cross-ratio is a mathematical invariant that describes the spatial relationships between four points in an image. It's a scalar value that remains unchanged under projective transformations, such as rotations, translations, and scaling.



$$\frac{\|\mathbf{P}_3 - \mathbf{P}_1\| \|\mathbf{P}_4 - \mathbf{P}_2\|}{\|\mathbf{P}_3 - \mathbf{P}_2\| \|\mathbf{P}_4 - \mathbf{P}_1\|}$$

Fig 2.5 Cross Ratio

The cross-ratio is used to match points between images, enabling tasks like stereo vision, structure from motion, and object recognition. The image cross-ratio is a powerful tool for describing the relationships between points in an image, and its invariance and uniqueness properties make it a fundamental component of many computer vision algorithms.

2.6 Height calculation

Cross ratio is projective invariant quantity which means it does not change under projective transformations. Cross ratio of the real word and the image can be compared and to find the height of the main object (see fig 2.6).

$$\frac{\|\mathbf{T}-\mathbf{B}\|\,\|\infty-\mathbf{R}\|}{\|\mathbf{R}-\mathbf{B}\|\,\|\infty-\mathbf{T}\|}=\frac{H}{R}$$

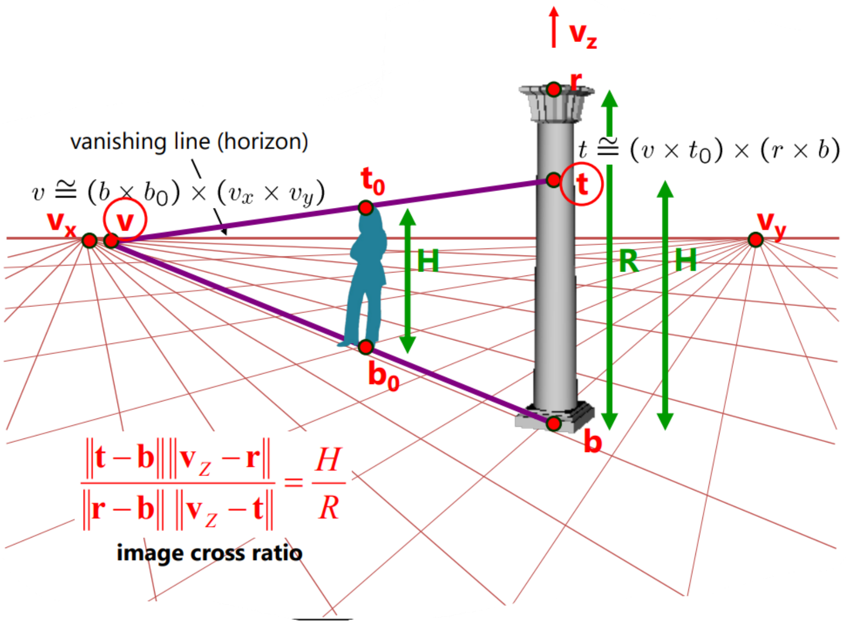
scene cross ratio

$$\frac{\|\mathbf{t}-\mathbf{b}\|\,\|\mathbf{v}_Z-\mathbf{r}\|}{\|\mathbf{r}-\mathbf{b}\|\,\|\mathbf{v}_Z-\mathbf{t}\|}=\frac{H}{R}$$

image cross ratio

From the real-world scene, the height of the reference object (H_r) will be known and the height of the main object will be unknown (R_r).

$$R_r = H_r \times (H_i/R_i)$$



$\mathbf{v_z}$
r
vanishing line (horizon)
$v \cong (b \times b_0) \times (v_x \times v_y)$
$t \cong (v \times t_0) \times (r \times b)$
$\mathbf{v_x}$
v
$\mathbf{t_0}$
t
$\mathbf{v_y}$
H
R
H
$\mathbf{b_0}$
b
$\frac{\|\mathbf{t}-\mathbf{b}\|\,\|\mathbf{v}_Z-\mathbf{r}\|}{\|\mathbf{r}-\mathbf{b}\|\,\|\mathbf{v}_Z-\mathbf{t}\|}=\frac{H}{R}$
image cross ratio

Fig 2.6 Labeled image of height measurement setup

Similarly, the width can also be estimated by implementing the above-mentioned method which requires a horizontal reference object in order to calculate the vanishing points.

2.7 Application

The validity of the metrology algorithm presented is demonstrated in this section with a number of practical applications.

2.7.1 Forensic science

A common requirement in surveillance images is to obtain measurements from the scene, such as the height of a suspect. Although, the suspect has usually departed the scene, reference lengths can be measured from fixtures such as tables and windows.

2.7.2 Defence application

In defence there are specific areas where there is need of object's dimension estimation for testing and evaluation purpose.

Since many defence inventory items requires rigorous testing in order to maintain their quality there is a need of metrological analysis so this method can be implemented which facilitates the process of measurement.

2.7.3 Mechanical arms in Industries

In industrial applications real time measurement of object is required since the mechanical arms deployed in industries for picking objects need tracking of specific objects on the conveyor this method can be deployed in order to track and measure object's dimensions.

2.8 Limitations

- Presence of object to find vanishing points or horizon should be visible.
- A reference object should be present whose dimensions are known.
- The reference object and main object should be parallel or near to parallel.
- A base object should be available for estimation of vanishing point Vx and Vy.
- Vanishing lines of 3D world should not be parallel with the image coordinate plane.

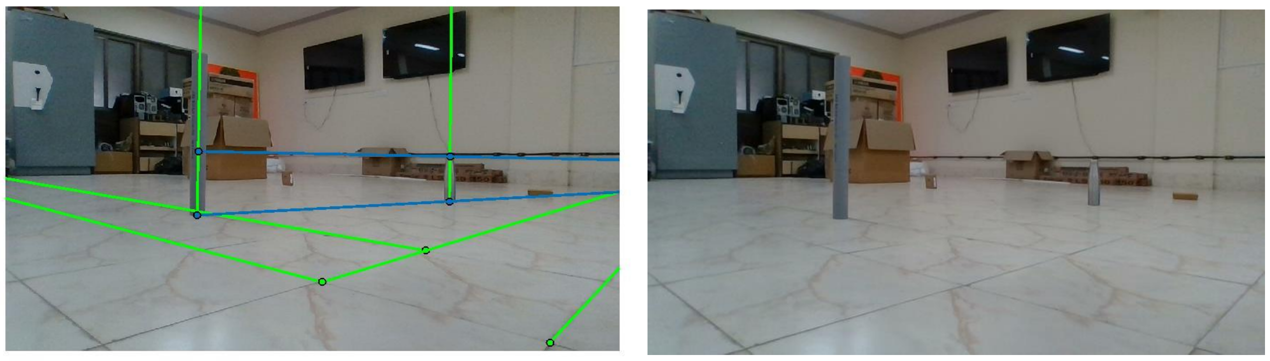
Chapter 3

Results

3.1 Estimation of object's Dimensions

This method was tested on multiple image and video data. MATLAB is used as an image processing environment tool. The image frames are extracted from video data and then segmentation method is used to track the main object in the frame.

Extrema are detected to get the end points of the segmented object. Then these end points are taken to calculate the vanishing points and image cross ratio which are used to estimate the dimensions of the main object.



A. Labeled Image B. Actual Image

Fig 3.1 Calculating height of bottle (here pipe is taken as reference object)

Co-ordinate	Vz	Vy	Vx	Vo	Calculated Height	Actual Height
X	207	-450.92	1487.8	2133.6	25.8893 cm	25 cm
Y	68008	282.0477	324.4	338.5		
Z	1	1	1	1		

Table 3.1 Vanishing points obtained from the experiment with Calculated and actual height

S. No	Actual (in cm)	Predicted (in cm)	Accuracy (in %)
1	12	11.8524	98.7696
2	50	52.4124	95.1753
3	64.8	67.3142	96.1201
4	41	44.7216	90.9229

Table 3.2 Accuracy calculation

The implemented single-view metrology method achieved an accuracy ranging from [90-98%]. This demonstrates the effectiveness of the approach in extracting object dimensions from non-perpendicular camera viewpoints. Further research can explore methods to improve accuracy, such as incorporating camera calibration for specific applications or refining the HSV segmentation algorithm for better object isolation in complex scenes.

Chapter 4

Conclusion

In conclusion, this project has successfully demonstrated the feasibility of a robust single view metrology approach for determining height from a single image or video frame, without relying on any external or intrinsic camera characteristics. The proposed method, implemented in MATLAB, offers a flexible solution that can be applied in various scenarios, including those where the camera is not positioned perpendicularly to the object. By utilizing HSV-based image segmentation, our approach can effectively handle objects with changing shapes, as commonly encountered in video sequences.

The significance of this work lies in its ability to provide accurate height measurements from a single view, without requiring prior knowledge of the camera's intrinsic or extrinsic properties. This makes it a valuable tool for a wide range of applications, including computer vision, robotics, and surveillance, where accurate object measurement is crucial.

Future work can focus on extending this approach to handle more complex scenarios, such as objects with varying textures or lighting conditions. Additionally, exploring the integration of this method with other computer vision techniques, such as object tracking or 3D reconstruction, could lead to even more powerful applications.

Overall, this project has contributed to the advancement of single view metrology in videos, and its outcomes have the potential to benefit various fields where accurate object measurement is essential.

Bibliography

[1] R. Hartley, Multiple View Geometry in Computer Vision, 2nd ed. Canberra, Australia: Australian National University, ISBN 0-521-54051-8, 2003.

[2] A. Criminisi, Accurate Visual Metrology from Single and Multiple Uncalibrated Images, Distinguished Dissertations. ISBN 978-1-4471-1040, 2001.

[3] Kang, H.-C.; Han, H.-N.; Bae, H.-C.; Kim, M.-G.; Son, J.-Y.; Kim, Y.-K. "HSV Color-Space-Based Automated Object Localization for Robot Grasping without Prior Knowledge." Appl. Sci. 2021, 11, 7593.

[4] Mundy, J.L. and Zisserman, A., Geometric Invariance in Computer Vision, Appendix: Projective Geometry for Machine Vision, MIT Press, Cambridge, MA, 1992

[5] M. Berger. Geometry II. Springer-Verlag, 1987.

[6] R. T. Collins and R. S. Weiss. Vanishing point calculation as a statistical inference on the unit sphere. In Proc. 3rd International Conference on Computer Vision, Osaka, pages 400–403, December 1990.

[7] A Criminisi, I. Reid, and A Zisserman. A plane measuring device. Image and Vision Computing, 17(8):625–634, 1999.

[8] A. Criminisi, I. Reid, and A. Zisserman. Single view metrology. In Proc. 7th International Conference on Computer Vision, Kerkyra, Greece, pages 434–442, September 1999.

[9] A. Criminisi, A. Zisserman, L. Van Gool, Bramble S., and D. Compton. A new approach to obtain height measurements from video. In Proc. of SPIE, Boston, Massachussets, USA, volume 3576, 1-6 November 1998.

[10] F. Devernay and O. D. Faugeras. From projective to euclidean reconstruction. In Proceedings of the Conference on Computer Vision and Pattern Recognition, pages 264–269, 1996.

[11] D. Huynh. Affine reconstruction from monocular vision in the presence of a symmetry plane. In Proc. 7th International Conference on Computer Vision, Kerkyra, Greece, September 1999.

Appendix

MATLAB Code for detecting the object height in video.

```
videoReader = VideoReader('roked_high.mp4');
videoReader.CurrentTime = 5.0;
videoPlayer = vision.VideoPlayer;

x2=[1710,709,1];
x5=[864,961,1];
f1=[1316,740,1];
f4=[1526,761,1];
f3=[864,961,1];
f2=[636,923,1];
v6=cross(f1,f4);
v7=cross(f2,f3);
v8=cross(v6,v7);

v9=(cross(f4,f3));
v10=(cross(f2,f1));
v11=(cross(v10,v9));
vx=[v11(1)/v11(3),v11(2)/v11(3),1];
vy=[v8(1)/v8(3),v8(2)/v8(3),1];
while hasFrame(videoReader)

    frame = readFrame(videoReader);
    mkdir = createMask(frame);
    mkdir = imfill(mkdir,'holes');
    mkdir = bwareaopen(mkdir,20);
    labeledimage = bwlabel(mkdir);
    measure=regionprops(labeledimage,'BoundingBox','Area');
    T = struct2table(measure);
    [m,i]=max(T.Area);
    measure123 = regionprops(labeledimage,'Extrema');
    subplot(1,1,1);
    imshow(frame);
    x3=[measure123(i).Extrema(7,:),1];
    x4=[measure123(i).Extrema(3,:),1];
    v3=(cross(x2,x5));
    v4=(cross(x4,x3));
    vz=(cross(v3,v4));
    vm=[vz(1)/vz(3),vz(2)/vz(3),1];
    va = cross(x2,x4);
    vb = cross(vx,vy);
    vt=cross(va,vb);
    v=[vt(1)/vt(3),vt(2)/vt(3),1];
    t=cross((cross(v,x3)),(cross(x5,x2)));
    t1=[t(1)/t(3),t(2)/t(3),1];
    roi3=images.roi.Line(gca,'Position',[x2(1) x2(2);vm(1)…
    vm(2) ],'color','green');
    roi4 = images.roi.Line(gca,'Position',[x4(1)…
    x4(2);vm(1) vm(2)],'color','green');
    roi11 = images.roi.Line(gca,'Position',[f2(1)...
    f2(2);v8(1)/v8(3) v8(2)/v8(3)],'color','green');
    roi12 = images.roi.Line(gca,'Position',[f1(1)…
    f1(2);v8(1)/v8(3) v8(2)/v8(3) ],'color','green');
    roi13 = images.roi.Line(gca,'Position',[f3(1) …
    f3(2);v11(1)/v11(3) v11(2)/v11(3) ],'color','green');
    roi14= images.roi.Line(gca,'Position',[f2(1) …
    f2(2);v11(1)/v11(3) v11(2)/v11(3) ],'color','green');
    roi17= images.roi.Line(gca,'Position',[t(1)/t(3)…
    t(2)/t(3);vt(1)/vt(3) vt(2)/vt(3)]);
    roi18= images.roi.Line(gca,'Position',[x2(1)x2(2);…
    v(1)v(2)]);
    roi15= images.roi.Line(gca,'Position',[x3(1)…
    x3(2);vt(1)/vt(3) vt(2)/vt(3)]);
    roi16= images.roi.Line(gca,'Position',[x4(1) …
    x4(2);vt(1)/vt(3) vt(2)/vt(3)]);
    l=magn(t1-x2)*magn(vz-x5);
    p=(magn(x5-x2)*magn(vz-t1));
    refheight=462;
    h=(l/p)*refheight;
    disp(h);
     lm=h;
     frame=insertObjectAnnotation(frame,'circle',x3,lm,'Font
     Size',19,'color','blue','LineWidth',6);
     frame=insertObjectAnnotation(frame,'circle',x4,"",'colo
     r','blue','LineWidth',5);
    step(videoPlayer,frame);
    drawnow;
    hold on;
end
release(videoPlayer);
```

```
function [BW,maskedRGBImage] = createMask(RGB)
I = rgb2hsv(RGB);
% Define thresholds for channel 1 based on histogram
settings
channel1Min = 0.979;
channel1Max = 0.216;
% Define thresholds for channel 2 based on histogram
settings
channel2Min = 0.000;
channel2Max = 0.457;
% Define thresholds for channel 3 based on histogram
settings
channel3Min = 0.992;
channel3Max = 1.000;
% Create mask based on chosen histogram thresholds
sliderBW = ( (I(:,:,1) >= channel1Min) | (I(:,:,1) <=
channel1Max) ) & ...
    (I(:,:,2) >= channel2Min ) & (I(:,:,2) <= channel2Max)
& ...
    (I(:,:,3) >= channel3Min ) & (I(:,:,3) <= channel3Max);
BW = sliderBW;
% Initialize output masked image based on input image.
maskedRGBImage = RGB;
% Set background pixels where BW is false to zero.
maskedRGBImage(repmat(~BW,[1 1 3])) = 0;
end
function mag = magn(a)
     mag=sqrt(a(1)^2+a(2)^2);
end
```