CHAPTER 2

Principles and theories of the study

This chapter describes the principles and theories related to the eye gaze tracking system design. Section 2.1 explains the fundamentals of eye gaze tracking system. Section 2.2 describes eye gaze image filtering using two-dimensional Gaussian filter. After the face image is filtered, both eye's positions can be determined by using the image labelling method described in section 2.3.

Before computing the eye gaze point on the screen, the distance between the participant's eye and the screen has to be determined. Thus, regression methods are described for the eye gaze distance estimation in section 2.4.

To compute the eye gaze point on the screen, the perspective transformation is used for image transformations as described in section 2.5. The eye gaze center can be determined using the eigenvalue decomposition as shown in section 2.6. Then, the iris radius used for a three-dimensional eye model is computed as described from section 2.7 through section 2.10.

Finally, a confidence interval is applied to verify the experimental results demonstrated in section 2.11.

2.1 Fundamental of eye gaze tracking

An eye gaze tracking system consists of the eye detection and tracking. The eye gaze tracking involves two areas according to the eye localization in the image and gaze estimation [35]. Also, the eye detection has three processes: First, the location of the eye on the facial image is detected; second, the position of the eye gaze in the image is computed and the points of gaze are interpreted as the screen positions; third, the eye gaze points are tracked via frame to frame from the images obtained by the video camera.

Then, the relationship between the iris center and glint positions, which was utilized for mapping the eye gaze vector to the screen positions, is calculated. The process of the eye gaze system is to find out where the user is looking at the 3-D point on the screen. The calculation of the point of gaze on the screen is called 'gaze estimation'.

The eye gaze estimation on the screen was performed by determining the eye location in the facial image. Then, the background color of the facial image was eliminated by the 2-D Gaussian filter. After that, the Blob coloring was used for labelling the point of glint in the image. The distance between the eye and the screen could be estimated using the linear and exponential regression methods. The human eye model required parameters which consist of the iris radius, eye gaze center, and glint positions. All eye parameters were used for eye gaze mapping to the screen positions. Figure 2.1 shows the process of the eye gaze tracking system.

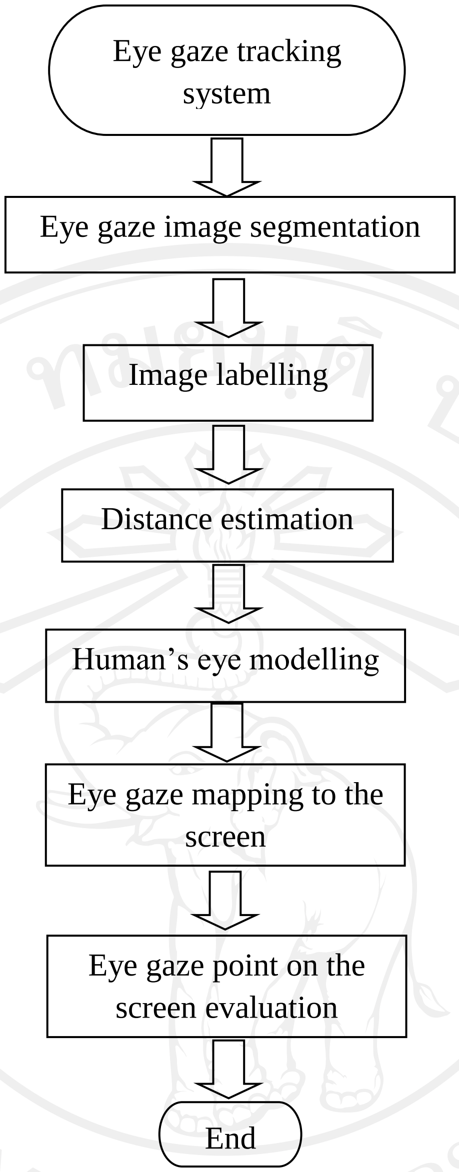


Figure 2.1 Process of eye gaze tracking system.

2.2 Eye gaze image filtering

The process of the eye gaze tracking system began with the eye gaze image segmentation. Both eyes in the image were required to be segmented. In order to extract the eye gaze image from the facial image, the two-dimensional (2-D) Gaussian filter method [36] was applied. The advantage of using the 2-D Gaussian filter is that it could subtract the undesirable facial features such as eyebrows, nose, and skin color. The theory of the two-dimensional Gaussian filter is described as follows.

2.2.1 Two-dimensional Gaussian filter

An image was smoothened by using the two-dimensional (2-D) Gaussian filter. The Gaussian smoothing operator $G(x,y)$ [36] or a Gaussian filter is given by

$$G(x,y) = e^{-(x^2+y^2)/2\sigma^2}. \tag{2.1}$$

where x, y are the image co-ordinates and σ is a standard deviation of the associated probability distribution. The Gaussian filter can also be represented by including normalizing factor as follows:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{2.2}$$

or

$$G(x,y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x^2+y^2)/2\sigma^2}. \tag{2.3}$$

The second derivative of a smoothing 2-D function $f(x, y)$ can be obtained by using the Laplace operator ∇^2. The Laplacian of an image $f(x, y)$ can be smoothened by using a convolution ($*$). Thus, the operation from Laplacian of Gaussian is defined as

$$\nabla^2 \left[G(x,y,\sigma) * f(x,y) \right]. \tag{2.4}$$

The expression in (2.4) can be interchanged to perform the differentiation and convolution as follows:

$$\left[\nabla^2 G(x,y,\sigma) \right] * f(x,y). \tag{2.5}$$

The complexity of the expression in (2.1) can be reduced by substituting $r^2=x^2+y^2$, where r measures the distance from the origin. The aim is to convert the 2-D Gaussian into 1-D function which is simpler to be differentiated as follows:

$$G(r) = e^{-r^2/2\sigma^2}. \tag{2.6}$$

The first derivative $G^{'}(r)$ is defined as

$$G'(r) = -\frac{1}{\sigma^2} r e^{-r^2/2\sigma^2} . \tag{2.7}$$

The second derivative $G''(r)$, the Laplacian of a Gaussian, is defined as

$$G''(r) = \frac{1}{\sigma^2}\left(\frac{r^2}{\sigma^2} - 1\right) e^{-r^2/2\sigma^2} . \tag{2.8}$$

After returning to the original co-ordinates *x*, *y* and introducing a normalizing multiplicative coefficient *c*, the convolution mask of a LoG operator is defined as

$$h(x, y) = c\left(\frac{x^2 + y^2}{\sigma^4}\right) e^{-\left(x^2+y^2\right)/2\sigma^2} . \tag{2.9}$$

The inverted LoG operator is commonly called a Mexican hat. An example of a 5×5 discrete approximation is

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}.$$

After the background of the facial image is filtered by using the two-dimensional Gaussian filter, both eye positions within the image can be determined. Then, an image labelling method is used to define both eye's positions. The method for image labelling is described in the next section.

2.3 Image labelling

The eye positions in an image could be determined by using the image labelling method. The Blob coloring was used for labelling both eyes positions in the facial image. The Blob coloring [37] was also applied for labelling the position of the point light sources reflected on the participant's cornea. The 4-connected component could be determined by using the mask as presented in Figure 2.2.

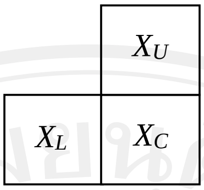


Figure 2.2 Blob coloring mask.

The Blob coloring method can be described as follows:

1) *Given the begin color is $k = 1$, then scanning the image from left to right and from top to the bottom.*
2) *If $f(X_C) = 0$, to be continuous for scanning the image*
3) *Else*

If $f(X_U) = 1$ and $f(X_L) = 0$

Color X_C = Color X_U

If $f(X_L) = 1$ and $f(X_U) = 0$

Color X_C = Color X_L

If $f(X_L) = 1$ and color $f(X_U) = 1$

Color X_C = Color X_U

Color X_L is equivalent to color X_U

If $f(X_L) = 0$ and $f(X_U) =0$

Color $X_C = k$

$k=k+1$

End

The Blob coloring was applied for locating the right and left eye positions. Then, the eyes were segmented from the face image. The segmented eye images were used for computing the eye gaze distance estimation between the participant's head and the screen. First of all, the eye gaze distance equation was computed by using the image patch which was reflected on the cornea. Moreover, the iris radius which was used for three-dimensional eye modelling was determined. The mathematical calculation associated with eye gaze distance estimation model is described in the next section.

2.4 Linear regression

2.4.1 Linear least squares

The linear least squares regression was utilized to find an approximation of the function that passed through the point of data obtained from an experiment [38]. The aim to use the least squares method is to find a function $f(x)$ of the eye gaze distance equation that best represented the data which was subjected to error. The linear function is generally presented as follows:

$$f(x) = ax + b. \tag{2.10}$$

The function $f(x)$ in (2.10) consists of parameters a and b that can make $f(x)$ be the "best" function which was used for fitting the data. Let $e_i = f(x_i) - y_i$ for $1 \leq i \leq n$. Therefore, the sum of square error is described as follows:

$$\begin{aligned} E(a,b) &= \sum_{i=1}^{n} (e_i)^2 \\ &= \sum_{i=1}^{n} \left[f(x_i) - y_i \right]^2 \\ &= \sum_{i=1}^{n} \left[(ax_i + b) - y_i \right]^2. \end{aligned} \tag{2.11}$$

The minimum of (2.11) will occur when

$$\frac{\partial E(a,b)}{\partial a} = 0, \text{ and } \frac{\partial E(a,b)}{\partial b} = 0. \tag{2.12}$$

This condition gives the following relations:

$$E(a,b) = \sum_{i=1}^{n} \left[(ax_i + b) - y_i \right]^2. \tag{2.13}$$

$$\frac{\partial}{\partial a}\sum_{i=1}^{n}\left[(ax_i+b)-y_i\right]^2 = 2\sum_{i=1}^{n}(ax_i+b-y_i)(x_i)=0$$
$$\frac{\partial}{\partial b}\sum_{i=1}^{n}\left[(ax_i+b)-y_i\right]^2 = 2\sum_{i=1}^{n}(ax_i+b-y_i)=0 \qquad (2.14)$$

The normal equation of least squares is

$$a\sum_{i=1}^{n}x^2{}_i+b\sum_{i=1}^{n}x_i=\sum_{i=1}^{n}x_iy_i\ ,$$

$$a\sum_{i=1}^{n}x_i+bn=\sum_{i=1}^{n}y_i\ . \qquad (2.15)$$

The solution of linear system is

$$a=\frac{n\sum x_iy_i-\sum x_i\sum y_i}{n\sum x_i{}^2-\left(\sum x_i\right)^2},$$

$$b=\frac{\sum x_i{}^2\sum y_i-\sum x_iy_i\sum x_i}{n\sum x_i{}^2-\left(\sum x_i\right)^2}\ . \qquad (2.16)$$

where $\sum=\sum_{i=1}^{n}$.

The linear regression was used to construct an equation estimating the linear data. However, in other cases, the data might not always be linear. Thus, linear formulas can possibly cause an error for distance estimation. The non-linear approach was conducted to compute the equation for estimating the function of the data set as described as follows:

2.4.2 Least squares polynomial

In the case of experimental data being non-linear, the least squares polynomial could be a choice for data fitting. The expression of the least squares polynomial is illustrated as follows:

$$p_m(x)=\sum_{k=0}^{m}a_kx^k\ . \qquad (2.17)$$

The coefficients a_0, a_1, … , a_m are required to find the minimum value of

$$E\left(a_0,...,a_m\right)=\sum_{i=1}^{n}\left[p_m\left(x_i\right)-y_i\right]^2$$
$$=\sum_{i=1}^{n}\left[\sum_{k=0}^{m}\left(a_k x_i^{\,k}\right)-y_i\right]^2. \qquad (2.18)$$

Fundamentally, E is minimum if

$$\frac{\partial}{\partial a_j}E\left(a_0,...,a_m\right)=0;\ j=0,1,...,m \qquad (2.19)$$

that is

$$\begin{aligned}
\frac{\partial E}{\partial a_0}&=\sum_{i=1}^{n}2\left[\sum_{k=0}^{m}\left(a_k x_i^{\,k}\right)-y_i\right]=0\\
\frac{\partial E}{\partial a_1}&=\sum_{i=1}^{n}2\left[\sum_{k=0}^{m}\left(a_k x_i^{\,k}\right)-y_i\right]\left(x_i\right)=0\\
.&=.\\
.&=.\\
\frac{\partial E}{\partial a_m}&=\sum_{i=1}^{n}2\left[\sum_{k=0}^{m}\left(a_k x_i^{\,k}\right)-y_i\right]\left(x_i^{\,m}\right)=0
\end{aligned} \qquad (2.20)$$

Rearranging (2.20) gives (m+1) normal equations with (m+1) unknowns, a_0, a_1,.., a_m , as shown below:

$$\begin{array}{ccccccccc}
a_0 n & + & a_1\sum x_i & +...+ & a_m\sum x_i^{\,m} & = & \sum y_i\\
a_0\sum x_i & + & a_1\sum x_i^{\,2} & +...+ & a_m\sum x_i^{\,m+1} & = & \sum y_i x_i\\
a_0\sum x_i^{\,2} & + & a_1\sum x_i^{\,3} & +...+ & a_m\sum x_i^{\,m+2} & = & \sum y_i x_i^{\,2}\\
. & + & . & +...+ & . & = & .\\
. & + & . & +...+ & . & = & .\\
. & + & . & +...+ & . & = & .\\
a_0\sum x_i^{\,m} & + & a_1\sum x_i^{\,m+1} & +...+ & a_m\sum x_i^{\,2m} & = & \sum y_i x_i^{\,m}
\end{array} \qquad (2.21)$$

The advantage of the least squares polynomial is that it can be used for non-linear regression. However, when dealing with experimental data which are exponential, another form of regression called exponential regression may be more suitable.

2.4.3 Exponential form

Even though the least squares polynomial can be utilized for non-linear data fitting, it depends on the nature of data. When the data in the experiment tend to possess an exponential pattern, this method may not be suitable. Instead, the exponential regression equation was used for data fitting. The exponential expression used for approximating the non-linear function is represented as follows:

$$f(x) = ae^{bx}. \tag{2.22}$$

In order to simplify the equation, the logarithmic of $f(x)$ is taken.

$$\ln f(x) = bx + \ln a. \tag{2.23}$$

If we let $F(x) = \ln f(x)$, $\alpha = \ln a$, and $\beta = \mathrm{b}$, then the linear function of x can be obtained as

$$F(x) = \beta x + \alpha. \tag{2.24}$$

The relation between β and α with the linear least squares is as follows:

$$b = \beta \text{ and } a = e^{\alpha}.$$

The theories described above were used for eye gaze distance estimation. Then, the distance between the participant's head and the screen was calculated. The estimated distance could be used for modelling a three-dimensional eye model of the human eye, which was the real size of the eye in the real-world coordinates. The method which was used for transforming the distance in the image into the distance in the real-world coordinates can be derived using the perspective transformation described in the next section.

2.5 Perspective transformation

The eye gaze tracking system needed to transform the 3-D eye gaze point in the real-world coordinate onto the 2-D plane. A perspective transformation [37] was applied to project 3-D points of the real-world coordinate onto image plane. A model of image formation process is shown in Figure 2.3.

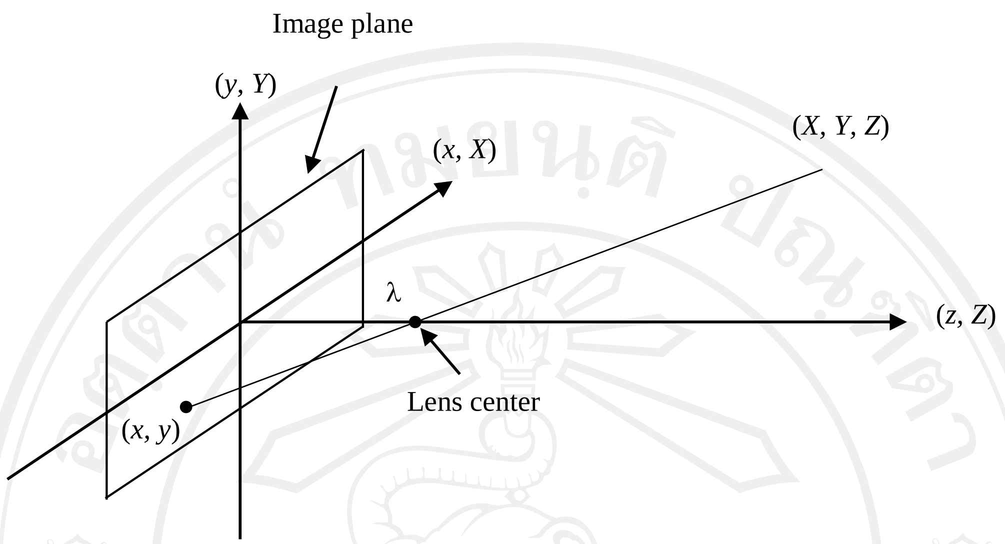


Figure 2.3 Model of the imaging process.

As shown in Figure 2.3, the *xy*-plane is the camera coordinate system (x, y, z) and the z-axis is the optical axis. The center of the image plane is set to the origin and the coordinate (0, 0, λ) is the center of the lens. The focal length of the lens, a distance where a camera is in focus for distance objects is λ.

It is assumed that the camera coordinate system aligns with the real-world one (X, Y, Z). From Figure 2.3, the real-world coordinate of any point in the 3-D scene is given as (X, Y, Z), assuming that $Z > \lambda$. All possible points of interest also lie in front of the lens. Here, the similar triangle can be utilized to project the point (X, Y, Z) onto the coordinates (x, y) of the image plane. The equation of similar triangles is presented as follows:

$$\frac{x}{\lambda} = -\frac{X}{Z-\lambda} = \frac{X}{\lambda - Z}\ , \tag{2.25}$$

and

$$\frac{y}{\lambda} = -\frac{Y}{Z-\lambda} = \frac{Y}{\lambda - Z}\ . \tag{2.26}$$

The image plane coordinates of the projected 3-D point are represented as follows:

$$x = \frac{\lambda X}{\lambda - Z}\ , \tag{2.27}$$

and

$$y = \frac{\lambda Y}{\lambda - Z}\ . \tag{2.28}$$

From (2.25) and (2.26), the image on the image plane is inverse to the 3-D point in the real-world coordinates. Both equations are non-linear expressions because they are divided by the variable Z. It is simpler for computation if this equation is rewritten in the matrix form. Here, it can be accomplished by using the homogeneous coordinates. The homogeneous coordinates of a point with Cartesian coordinates (X, Y, Z), are defined as (kX, kY, kZ, k) while the parameter k is an arbitrary nonzero constant. Thus, the transformation of the homogeneous coordinates can be obtained by dividing the first of three members with the fourth member. Therefore, the point in the Cartesian coordinate system is presented in the vector form as follows:

$$\mathbf{w} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2.29}$$

The homogeneous counterpart is given by

$$\mathbf{w}_h = \begin{bmatrix} kX \\ kY \\ kZ \\ k \end{bmatrix}. \tag{2.30}$$

Therefore, the matrix of perspective transformation is defined as

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{\lambda} & 1 \end{bmatrix}. \tag{2.31}$$

where $\mathbf{c}_h$ is obtained from the product $\mathbf{Pw}_h$, i.e.

$$\mathbf{c}_h = \mathbf{Pw}_h \,. \tag{2.32}$$

$$\mathbf{c}_h = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\dfrac{1}{\lambda} & 1 \end{bmatrix} \begin{bmatrix} kX \\ kY \\ kZ \\ k \end{bmatrix} = \begin{bmatrix} kX \\ kY \\ kZ \\ -\dfrac{kZ}{\lambda} + k \end{bmatrix}. \tag{2.33}$$

From (2.33), the result can be converted to the Cartesian form by dividing the equation with the fourth components of $\mathbf{c}_h$.

$$\mathbf{c} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \dfrac{\lambda X}{\lambda - Z} \\ \dfrac{\lambda Y}{\lambda - Z} \\ \dfrac{\lambda Z}{\lambda - Z} \end{bmatrix}. \tag{2.34}$$

The projected 3-D points (X, Y, Z) onto the 2-D image plane are the first two components of $\mathbf{c}$ which are the (x, y) coordinates according to (2.25) - (2.26). From (2.33), the image plane can be mapped to the 3-D coordinate system by using inverse perspective transformation. The inverse perspective transformation is defined as follows:

$$\mathbf{w}_h = \mathbf{P}^{-1}\mathbf{c}_h \,. \tag{2.35}$$

where the matrix $\mathbf{P}^{-1}$ is defined as

$$\mathbf{P}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\dfrac{1}{\lambda} & 0 \end{bmatrix}. \tag{2.36}$$

Assume the image points have the coordinates (x_0, y_0, 0) with z = 0 which represents the image plane located at z = 0. Thus, the homogeneous vector form of $\mathbf{c}_h$ can be expressed as

$$\mathbf{c}_h = \begin{bmatrix} kx_0 \\ ky_0 \\ 0 \\ k \end{bmatrix} \tag{2.37}$$

where z = 0.

From (2.35), the homogeneous world coordinate vector can be demonstrated as

$$\mathbf{w}_h = \begin{bmatrix} kx_0 \\ ky_0 \\ 0 \\ k \end{bmatrix}. \tag{2.38}$$

In terms of the Cartesian coordinates, it can be defined as

$$\mathbf{w} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ 0 \end{bmatrix}. \tag{2.39}$$

Therefore, the result from (2.39) is the image plane transformation to the real-world coordinate that also causes Z = 0. The image point (x_0, y_0) is the set of co-linear 3-D points that lie on the line passing through (x_0, y_0, 0) and (0, 0, λ). Therefore, the real world coordinate system can be derived from (2.27) and (2.28) as follows:

$$X = \frac{x_0}{\lambda}(\lambda - Z),$$

$$Y = \frac{y_0}{\lambda}(\lambda - Z). \tag{2.40}$$

Equation (2.40) could be used for computing the real-world coordinate from an image when Z and λ are given. Next, in order to compute the eye gaze displacements, the eye gaze center has to be determined. The method used to locate the eye gaze center using the eigenvalue technique is discussed below.

2.6 Eigenvalue decomposition

Eigenvectors and eigenvalues are useful for eye gaze center detection. Here, consider the matrix equation of eigenvalue decomposition. If the matrix **C** is symmetric, it can be written as eigenvalues decomposition [39] as follows:

$$\mathbf{C} = U\Lambda U^T = \left[\begin{array}{c|c|c} u_0 & \cdots & u_{n-1} \end{array} \right] \begin{bmatrix} \lambda_0 & & \\ & \ddots & \\ & & \lambda_{n-1} \end{bmatrix} \left[\begin{array}{c} u_0^T \\ \hline \cdots \\ \hline u_{n-1}^T \end{array} \right],$$

$$= \sum_{i=0}^{n-1} \lambda_i u_i u_i^T \tag{2.41}$$

where

$$\lambda_0 \geq \lambda_1 \geq \ldots \geq \lambda_{n-1}. \tag{2.42}$$

The symmetric matrix **C** can be constructed as the sum of a number of outer products.

$$\mathbf{C} = \sum_i a_i a_i^T = \mathbf{A}\mathbf{A}^T \tag{2.43}$$

The matrix **A** includes all the a_i column vectors which are appended columnwise. In this case, all of the eigenvalues λ_i are non-negative. The associated matrix **C** is *positive semi-definite* if and only if

$$x^T \mathbf{C} x \geq 0, \ \forall x. \tag{2.44}$$

The covariance of a set of $\{x_i\}$ points around their mean $\bar{x}$ is defined as

$$\mathbf{C} = \frac{1}{n} \sum_i (x_i - \bar{x})(x_i - \bar{x})^T. \tag{2.45}$$

The eigenvalue decomposition is the *principal component analysis* (PCA). The eigenvalues, eigenvectors of **C,** the singular values, and the singular vectors of **A** are given by

$$\mathbf{A}=\boldsymbol{U}\boldsymbol{\Sigma}\mathbf{V}^{T} \tag{2.46}$$

and

$$\mathbf{C}=\mathbf{A}\mathbf{A}^{T}=\mathbf{U\Sigma V^{T}V\Sigma U^{T}}=\mathbf{U\Lambda U^{T}} \tag{2.47}$$

The $\lambda_i = \sigma_i^2$ and that the left singular vectors of **A** are the eigenvectors of **C**. The individual differences from mean image $a_i = x_i - \bar{x}$ are long vectors of length P (the number of pixels in the image), while the total number of exemplars N (the number of images in the training database) is much smaller. Instead of forming $\mathbf{C} = \mathbf{A}\mathbf{A}^T$ which is $P \times P$, **C** can be formed as an $N \times N$ matrix by

$$\hat{\mathbf{C}}=\mathbf{A}^{T}\mathbf{A}\,. \tag{2.48}$$

The eigenvectors of $\hat{\mathbf{C}}$ are the square singular values of **A,** namely $\sum^2$, and are hence also the eigenvalues of **C**. The eigenvectors $\hat{\mathbf{C}}$ are the right singular vectors **V** of **A**, from which the desired eigenfaces **U**, which are the left singular vector of **A**, can be compute as

$$\mathbf{U}=\mathbf{A}\mathbf{V}\boldsymbol{\Sigma}^{-1}\,. \tag{2.49}$$

The eigenvector can be defined as

$$\lambda_i u_i=\mathbf{C}u_i \;\; \textbf{or} \;\; \left(\lambda_i-\mathbf{C}\right)u_i=0 \tag{2.50}$$

The non-trivial solution for u_i can be founded if the system is rank deficient, i.e.,

$$\left|\lambda\mathbf{I}\text{-}\mathbf{C}\right|=0 \tag{2.51}$$

The determinant form the characteristic polynomial in λ, which can be solved for small problems, e.g., 2×2 or 3×3 matrices, in closed form.

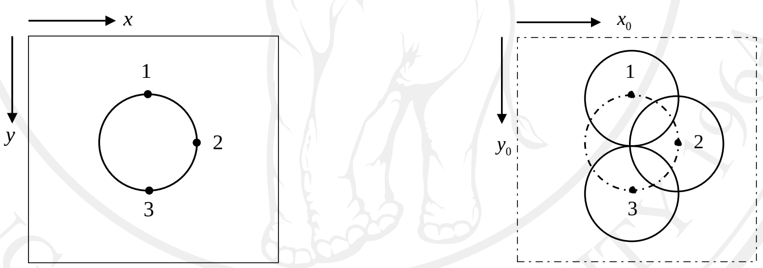
After the eye gaze center had been computed, it was required to determine the actual iris radius which needed for computing the visual eye angle. The visual eye angle was used in the process of eye gaze displacement mapping to the screen position. The Hough transform method could estimate the iris radius and was used for the proposed method, as described in the next section.

2.7 Hough transform for circle

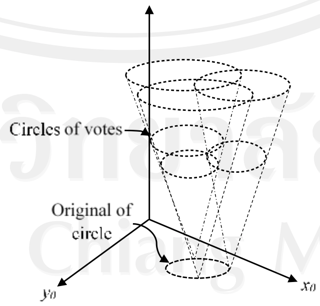
The Hough transform circle [41] can be considered from the equation of a circle which is defined as

$$(x-x_0)^2+(y-y_0)^2=r^2. \tag{2.52}$$

The locus of point (x, y) is centered on the origin (x_0, y_0) with radius r. The equation can be expressed in two ways: by the locus of point (x, y) in the image, and by the locus of point (x_0, y_0) centered on (x, y) with radius r. The example of Hough transform for circles is illustrated in Figure 2.4.



(a) Image containing a circle (b) Accumulator space



(c) 3-D accumulator space

Figure 2.4 Hough transform for circles.

Figure 2.4(a) shows a defined set of circles in accumulator space with all possible values of radius *r*. These centers of circles are in the coordinates of the edge point. Figure 2.4(b) shows three circles in accumulator space defined by edge points for a given radius value. Three edge points are mapped to be a cone of vote in the accumulator space as illustrated in Figure 2.4(c). The maximum in the accumulator space corresponds to the parameter of circle in the original image. Thus, the equation of HT (2.53) for a circle can be defined in parametric form as

$$x = x_0 + r\cos(\theta),\ \ y = y_0 + r\sin(\theta)\,. \qquad (2.53)$$

The Hough transform mapping is defined by

$$x_0 = x - r\cos(\theta),\ y_0 = y - r\sin(\theta)\,. \qquad (2.54)$$

The points in the accumulator space (Figure 2.4(b)) are dependent on the radius *r*, while θ is defined as the trace of the curve that refers to the point spread function.

2.8 Parameter space reduction for circles

There are different geometric properties of a circle to decompose the parameter space. In this case, the second directional derivative is used for parameter decomposition. The derivative equation is obtained by considering equation (2.53) that defines a position vector function. That is,

$$\omega(\theta) = x(\theta)\begin{bmatrix}1\\0\end{bmatrix} + y(\theta)\begin{bmatrix}0\\1\end{bmatrix} \qquad (2.55)$$

where

$$x(\theta) = x_0 + r\cos(\theta)\,,\ y(\theta) = y_0 + r\sin(\theta)\,. \qquad (2.56)$$

The first and second directional derivatives of (2.55) are defined as

$$v'(\theta) = x'(\theta)\begin{bmatrix}1\\0\end{bmatrix} + y'(\theta)\begin{bmatrix}0\\1\end{bmatrix}$$

$$v''(\theta) = x''(\theta)\begin{bmatrix}1\\0\end{bmatrix} + y''(\theta)\begin{bmatrix}0\\1\end{bmatrix} \qquad (2.57)$$

where

$$x'(\theta) = -r\sin(\theta)\ ,\ y'(\theta) = r\cos(\theta)$$
$$x''(\theta) = -r\cos(\theta)\ ,\ y''(\theta) = -r\sin(\theta). \tag{2.58}$$

Figure 2.5 shows the definition of the first and second directional derivatives for a circle. The first derivative defines a tangential vector while the second derivative is similar to the vector function.

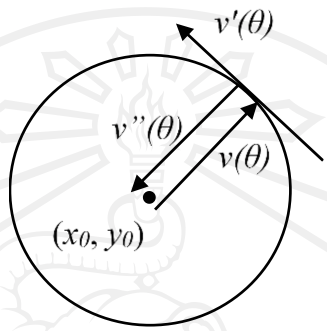


Figure 2.5 Definition of the first and second directional derivatives for a circle.

The tangent of the angle of the first directional derivative from (2.56) and (2.58) can be denoted as $\phi'(\theta)$ and is given by

$$\phi'(\theta) = \frac{y'(\theta)}{x'(\theta)} = -\frac{1}{\tan(\theta)}. \tag{2.59}$$

In the angular form, this equation can be written as

$$\hat{\phi}'(\theta) = \tan^{-1}\left(\phi'(\theta)\right). \tag{2.60}$$

The tangent of the second directional derivative is

$$\phi''(\theta) = \frac{y''(\theta)}{x''(\theta)} = \tan(\theta),$$

$$\hat{\phi}''(\theta) = \tan^{-1}\left(\phi''(\theta)\right). \tag{2.61}$$

The definition of $\phi''(\theta)$ is

$$\phi''(\theta) = \frac{y''(\theta)}{x''(\theta)} = \frac{y(\theta) - y_0}{x(\theta) - x_0}. \tag{2.62}$$

This equation defines a straight line that passes through the point $(x(\theta), y(\theta))$ and (x_0, y_0). The equation can be rearranged as

$$y(\theta) = \phi''(\theta)(x(\theta) - x_0) + y_0 . \tag{2.63}$$

From (2.63), the radius parameter is dependent. Therefore, it can be used to gather evidence of the location of the shape in a 2-D accumulator. Here, the Hough transform is also defined in the new form as

$$y_0 = \phi''(\theta)(x_0 - x(\theta)) + y(\theta) . \tag{2.64}$$

Thus, from given image point $(x(\theta), y(\theta))$ and the value of $\phi''(\theta)$, the line of vote in the 2-D accumulator (x_0, y_0) is generated. The purpose of the parameter space decomposition is to obtain the value of $\phi''(\theta)$ from image data.

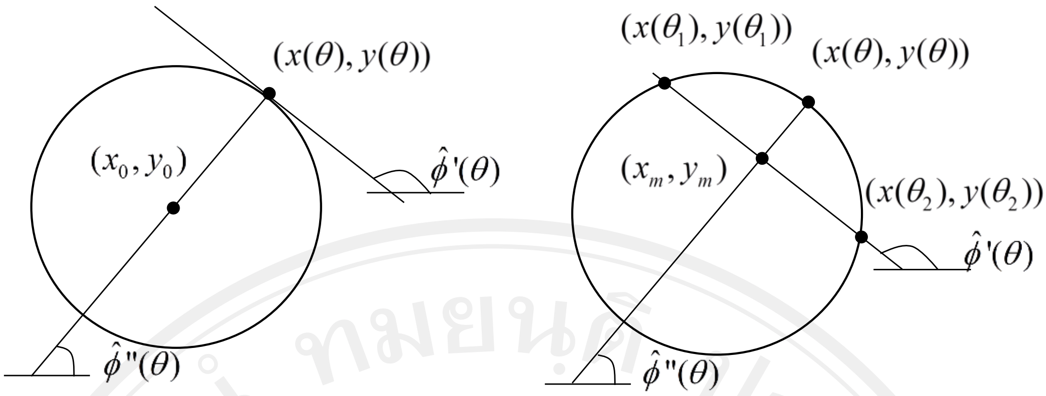
In order to obtain $\phi''(\theta)$, the equation (2.60) and (2.61) can be used for computation. The tangents of $\phi''(\theta)$ and $\phi'(\theta)$ are perpendicular. The relationship of equation can be represented as

$$\phi''(\phi) = -\frac{1}{\phi'(\theta)} . \tag{2.65}$$

The Hough transform in (2.64) can be written in terms of gradient direction $\phi'(\theta)$ as

$$y_0 = y(\theta) + \frac{x(\theta) - x_0}{\phi'(\theta)} . \tag{2.66}$$

This equation can be described as the line of votes passes through the point $(x(\theta), y(\theta))$ and (x_0, y_0) as shown in the Figure 2.6(a). The slope of the line is perpendicular to the direction of gradient direction.



(a) Relationship between angles. (b) Two point angle definition.

Figure 2.6 Geometry of the angle of the first and second directional derivatives.

There is an alternative method to obtain the parameter decomposition. From Figure 2.6(b), if a pair of points (x_1, y_1) and (x_2, y_2) is taken, where $x_i = x(\theta_i)$, then the lines passing through the points $(x(\theta), y(\theta))$ have the same slope. That is

$$\phi'(\theta) = \frac{y_2 - y_1}{x_2 - x_1} \tag{2.67}$$

where

$$\theta = \frac{1}{2}\left(\theta_1 + \theta_2\right). \tag{2.68}$$

The second directional derivative is

$$\phi''(\theta) = -\frac{x_2 - x_1}{y_2 - y_1}. \tag{2.69}$$

From (2.68), the location of the point $(x(\theta), y(\theta))$ is not known. However, the voting line also passes through the midpoint of the line between two selected points. Therefore, this point can be defined as

$$x_m = \frac{1}{2}(x_1 + x_2) \quad , \quad y_m = \frac{1}{2}(y_1 + y_2). \tag{2.70}$$

Thus, the Hough transform mapping can be written by substituting (2.67) into (2.66). In addition, by replacing the point $(x(\theta), y(\theta))$ with (x_m, y_m), the HT mapping can be expressed as follows:

$$y_0 = y_m + \frac{(x_m - x_0)(x_2 - x_1)}{(y_2 - y_1)}. \tag{2.71}$$

This equation is based on a pair of points that can be used for reducing the parameter space decomposition. In the case of a circle, tangents can be computed by gradient direction or by a pair of points.

2.9 Hough transform for ellipse

The problem of the camera's viewpoint is that circles do not always look like circles, especially in eye images where they possibly look like ellipses. Thus, in reality, images are formed by mapping a shape in 3-D space into the image plane. In other word, such function mapping performs a perspective transformation. A circle is deformed to look like an ellipse. The circle can be transformed to an ellipse by a similarity transformation [40],

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(\rho) & \sin(\rho) \\ -\sin(\rho) & \cos(\rho) \end{bmatrix} \begin{bmatrix} S_x & \\ & S_y \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}. \tag{2.72}$$

where (x', y') is defined as the coordinates of the circle in (2.53), ρ represents the orientation, (s_x, s_y) a scale factor, and (t_x, t_y) a translation. The parameter can be defined as

$$\begin{aligned} a_0 &= t_x,\ a_x = S_x \cos(\rho),\ b_x = S_y \sin(\rho) \\ b_0 &= t_y,\ a_y = -S_x \sin(\rho),\ b_y = S_y \cos(\rho) \end{aligned} \tag{2.73}$$

Then the ellipse equation is given by

$$\begin{aligned} x &= a_0 + a_x \cos(\theta) + b_x \sin(\theta) \\ y &= b_0 + a_y \cos(\theta) + b_y \sin(\theta) \end{aligned} \tag{2.74}$$

which is represented in the polar form containing six parameters (a_0, b_0, a_x, b_x, a_y, b_y). θ is not a free parameter and it only addresses a particular point in the locus of the ellipse. The center of the ellipse can be defined as (a_0, b_0) and three axis parameters (a_x, b_x, a_y, b_y). The axis parameters can be related to the orientation and the length along the axes by

$$\tan(\rho) = \frac{a_y}{a_x}\ ,\ a = \sqrt{a_x^{\ 2} + a_y^{\ 2}}\ ,\ b = \sqrt{b_x^{\ 2} + b_y^{\ 2}}\ . \tag{2.75}$$

where (a, b) is the axes of the ellipse, as illustrated in Figure 2.7.

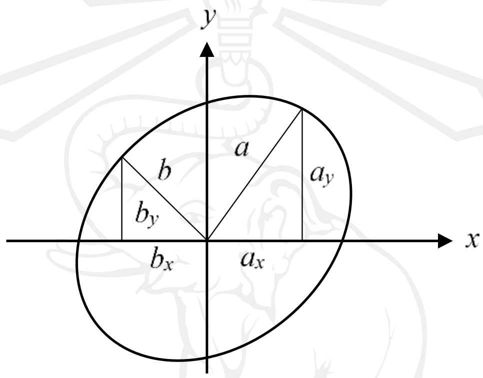


Figure 2.7 Ellipse axes.

Equation in (2.74) can be used to generate the mapping function for Hough transform for ellipses. Therefore, the location of the center of the ellipse is given by

$$\begin{aligned} a_0 &= x - a_x \cos(\theta) + b_x \sin(\theta) \\ b_0 &= y - a_y \cos(\theta) + b_y \sin(\theta) \end{aligned} \tag{2.76}$$

Which contain five dimensional (5-D) spaces of parameters. This can, therefore require large space for computation. However, the parameter reduction is required for HT mapping function.

2.10 Parameter space reduction for ellipses

It is difficult to reduce the parameter space for ellipses since ellipses have more free parameters and are geometrically more complex than circles. Thus, geometric properties of ellipse involve more complex relationships between points, tangents, and angles than those of circles. According to (2.55) and (2.57), the vector position and directional derivative of an ellipse in (2.74) can be written as

$$\begin{aligned} x'(\theta) &= -a_x \sin(\theta) + b_x \cos(\theta), \quad y'(\theta) = -a_y \sin(\theta) + b_y \cos(\theta) \\ x''(\theta) &= -a_x \cos(\theta) - b_x \sin(\theta), \quad y''(\theta) = -a_y \cos(\theta) - b_y \sin(\theta) \end{aligned} . \qquad (2.77)$$

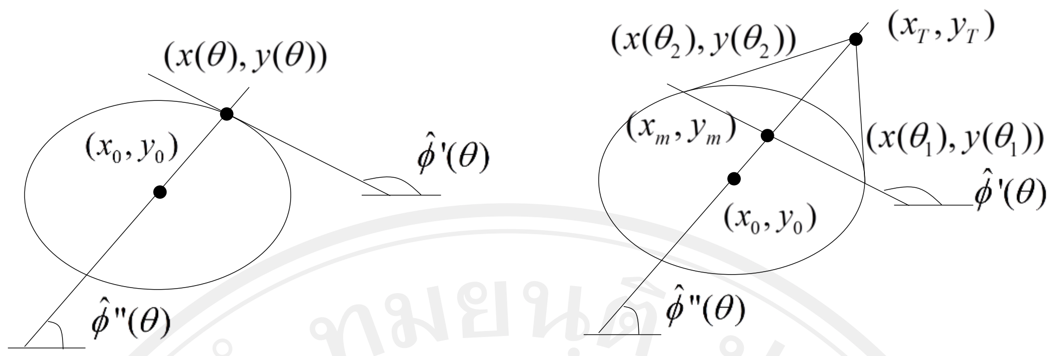
The tangents of the angles of the first and second directional derivatives are given by

$$\begin{aligned} \phi'(\theta) &= \frac{y'(\theta)}{x'(\theta)} = \frac{-a_y \cos(\theta) + b_y \sin(\theta)}{-a_x \cos(\theta) + b_x \sin(\theta)} \\ \phi''(\theta) &= \frac{y''(\theta)}{x''(\theta)} = \frac{-a_y \cos(\theta) - b_y \sin(\theta)}{-a_x \cos(\theta) - b_x \sin(\theta)} \end{aligned} . \qquad (2.78)$$

Therefore, from (2.77), the second derivative $\phi''(\theta)$ is defined as

$$\frac{y(\theta) - y_0}{x(\theta) - x_0} = \phi''(\theta) . \qquad (2.79)$$

Figure 2.8(a) represents the geometry of the definition for this equation. In the case of circles, this equation defines a line that passes through the points $(x(\theta), y(\theta))$ and (x_0, y_0) whereas in the case of ellipses, the angles $\hat{\phi}'(\theta)$ and $\hat{\phi}''(\theta)$ are not orthogonal. This causes the computation of $\phi''(\theta)$ more complex. A pair of points presented in Figure 2.8(b) is utilized to define a line whose slope defines the value of $\phi''(\theta)$ at another point. The line in (2.79) passes through the center point (x_m, y_m). Nevertheless, it is not orthogonal to the tangent line. Then, equation of (2.68) is utilized to obtain the second derivative $\phi''(\theta)$.



(a) Relationship between angles (b) Two point angle definition

Figure 2.8 Geometry of the angle of the first and second directional derivatives.

Three points in Figure 2.8(b) can be defined by

$$\begin{aligned} x_1 &= a_x \cos(\theta_1)\,, \quad x_2 = a_x \cos(\theta_2)\,, \quad x(\theta) = a_x \cos(\theta) \\ y_1 &= b_x \sin(\theta)\,, \quad y_2 = b_x \sin(\theta_2)\,, \quad y(\theta) = b_x \sin(\theta)\,. \end{aligned} \tag{2.80}$$

The point $(x(\theta), y(\theta))$ is given by the intersection of the line in (2.79) with the ellipse.

$$\frac{y(\theta) - y_0}{x(\theta) - x_0} = \frac{a_x}{b_y} . \frac{y_m}{x_m} \tag{2.81}$$

By substituting of the values of (x_m, y_m) in (2.70),

$$\tan(\theta) = \frac{a_x}{b_y} . \frac{b_y \sin(\theta_1) + b_y \sin(\theta_2)}{a_x \cos(\theta_1) + a_x \cos(\theta_2)} . \tag{2.82}$$

Thus,

$$\tan(\theta) = \tan\left(\frac{1}{2}(\theta_1 + \theta_2)\right). \tag{2.83}$$

The tangent angle of the second directional derivative is defined as

$$\phi''(\theta) = \frac{b_y}{a_x} \tan(\theta)\,. \tag{2.84}$$

By substituting in (2.81), it is defined as

$$\phi''(\theta) = \frac{y_m}{x_m}. \tag{2.85}$$

If the ellipse is translated, the tangent of the angle can be written in terms of the points (x_m, y_m) and (x_τ, y_τ) as

$$\phi''(\theta) = \frac{y_T - y_m}{x_T - x_m}. \tag{2.86}$$

Since the point (x_τ, y_τ) is the intersection point of the tangent lines at (x_1, y_1) and (x_2, y_2), the tangent of the angle can be obtained as

$$\phi''(\theta) = \frac{AC + 2BD}{2A + BC}. \tag{2.87}$$

where

$$\begin{aligned} A &= y_1 - y_2 \ , \ B = x_1 - x_2 \\ C &= \phi_1 + \phi_2 \ , \ D = \phi_1 \cdot \phi_2 \end{aligned}. \tag{2.88}$$

The parameters of ϕ_1, ϕ_2 are the slopes of the tangent line to the points. As in (2.79), the Hough transform mapping for the center parameter is defined as

$$y_0 = y_m + \frac{AC + 2BD}{2A + BC}(x_0 - x_m). \tag{2.89}$$

The Hough ellipse transform was selected for determining the iris radius. After the eye gaze displacements had been computed, the visual angle and the distance between the participant's head and screen became known values. Then, the eye gaze point on the screen position could be computed. The accuracy of the eye gaze on the screen positions evaluated the performance of the proposed system. In addition, the system accuracy was claimed by computing the confidence interval of the eye gaze point on the screen. Therefore, in the next section, the theory of the confidence interval is described.

2.11 Confidence interval for μ of the normal distribution with unknown σ^2

A confidence interval is used for measuring an unknown parameter θ of some distribution ($\theta = \mu$) which are in the interval of $\theta_1 \le \theta \le \theta_2$ [41]. This method is not certain but has high probability γ. Generally, the interval between 95 % and 99 % is popular. For example, if the confidence interval γ = 95 %, then probability becomes $1 - \gamma$ = 5 % = 1/20, implying that one out of 20 cases does not contain θ. The confidence interval for θ is written as

$$\mathrm{CONF}_{\gamma}\left\{\theta_1 \leq \theta \leq \theta_2\right\}. \tag{2.90}$$

where γ is the confidence level, θ_1 and θ_2 are the lower and upper confidence limits.

The larger confidence level γ causes a smaller error $(1-\gamma)$. In the case that $\gamma \rightarrow 1$, the length then goes to infinity. From (2.90), the midpoint of θ can be approximated and half the length of (2.90) can be considered as an error bound. The parameters θ_1 and θ_2 are calculated from the sample $x_1, \ldots, x_n$. These samples are n observations of a random variable X. Thus, $\theta_1 = \theta_1(x_1, \ldots, x_n)$ and $\theta_2 = \theta_2(x_1, \ldots, x_n)$ in (2.90) are observation values of two random variables $\Theta_1 = \Theta_1(X_1, \ldots, X_n)$ and $\Theta_2 = \Theta_2(X_1, \ldots, X_n)$. The confidence level γ is defined as follows:

$$P(\Theta_1 \leq \theta \leq \Theta_2) = \gamma \tag{2.91}$$

If σ^2 is unknown, the method of confidence interval with known σ^2 cannot be used. Because the values of k differ according to the sample standard deviation (s) has been determined with unknown the standard deviation σ of the population. In addition, the c depends on the sample size n, and must be determined lists values z for given values of the distribution function of the t-distribution. The expression for determining c can be written as follows:

$$F(z) = K_m \int_{-\infty}^{z} \left(1 + \frac{u^2}{m}\right)^{-(m+1)/2} du\,. \tag{2.92}$$

Here, m = (1, 2, 3, ….) is the parameter, called the number of degrees of freedom of the distribution.

The steps for determining the confidence interval for the mean of a normal distribution with unknown variance σ^2 can be described as follows:

Step 1. Choose a confidence level γ (95%, 99%).

Step 2. Determine the solution c of the equation.

$$F(c) = \frac{1}{2}(1+\gamma), \tag{2.93}$$

from the table of the t-distribution with n-1 degree of freedom.

Step 3. Compute the mean $\bar{x}$ and the variance s^2 of the sample $x_1, ..., x_n$.

Step 4. Compute $k = cs/\sqrt{n}$. The confidence interval is

$$\text{CONF}_\gamma\{\bar{x} - k \le \mu \le \bar{x} + k\}, \tag{2.94}$$

Here is a numerical example for confidence interval computation for μ of the normal distribution with unknown σ^2.

Suppose five dependences of the point of inflammation (flash point) of Diesel oil (D-2) give the values (in °F), 144, 147, 146, 142, and 144. Determine the 99% confidence interval for the mean.

Solution.

Step 1. γ =0.99.

Step 2. $F(c) = \frac{1}{2}(1+\gamma) = 0.995$ and from table A9, with n–1=4 degrees of freedom gives c = 4.60.

Step 3. $\bar{x} = 144.6$, $s^2 = 3.8$.

Step 4. $k = \sqrt{3.8} \times 4.60/\sqrt{5} = 4.01$. The confidence interval is

$$\text{CONF}_{0.99}\ \{140.5 \le \mu \le 148.7\}.$$