Computer Vision Face Tracking for Use in a Perceptual User Interface

G. R. Bradski, Microcomputer Research Lab, Santa Clara, CA, Intel Corporation.
Intel Technical Journal, pages 1-15, 1998.

Keywords: Computer vision, Face tracking, Mean shift algorithm, Perceptual user interface, 3D graphics interface.

吳品頡 Pin-Jie Wu
f7497623@mail.ncku.edu.tw
2012/03/12

簡報承自鐘凱融學長

Outline

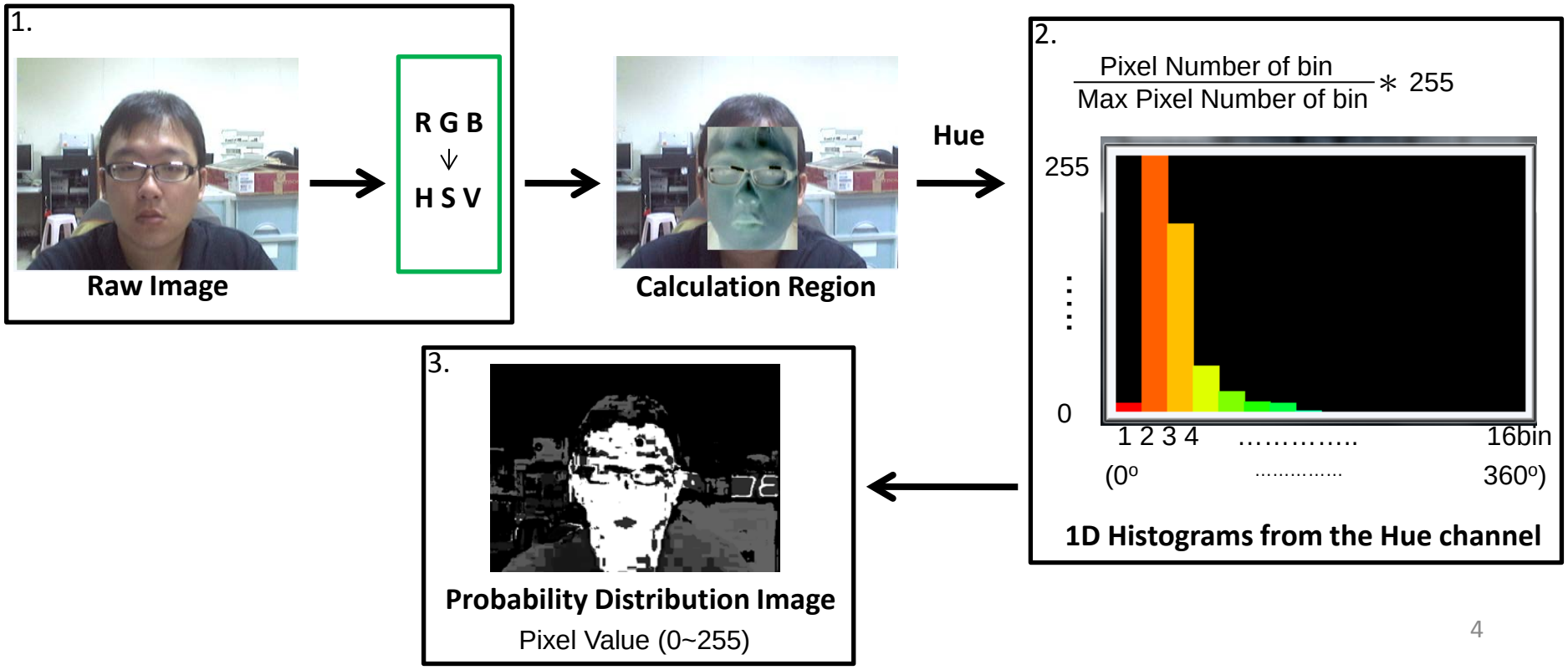
1. Introduction
2. System Flowchart
3. Color Probability Distribution
4. Mean Shift
5. CAMSHIFT
   - 5.1 Implementation Detail
   - 5.2 CAMSHIFT's Use As A Perceptual Interface
6. CAMSHIFT Analysis
7. Discussion
8. Conclusion
9. Reference

1. Introduction

- **Human Face Tracking:**
  - **Color-Based Tracking:** In order to find a fast, simple algorithm for basic tracking, we have focus on color-based tracking to track human faces.
  - **Statistics and Probability Distributions**: We use statistics and probability distributions to develop a fast, computationally efficient algorithm.
  - **Mean Shift**: We chose to use a robust non-parametric technique for climbing density gradients to find the mode of probability distributions.
  - **Continuously Adaptive Mean Shift (CAMSHIFT):** We use the new algorithm to track the dynamical probability distribution.

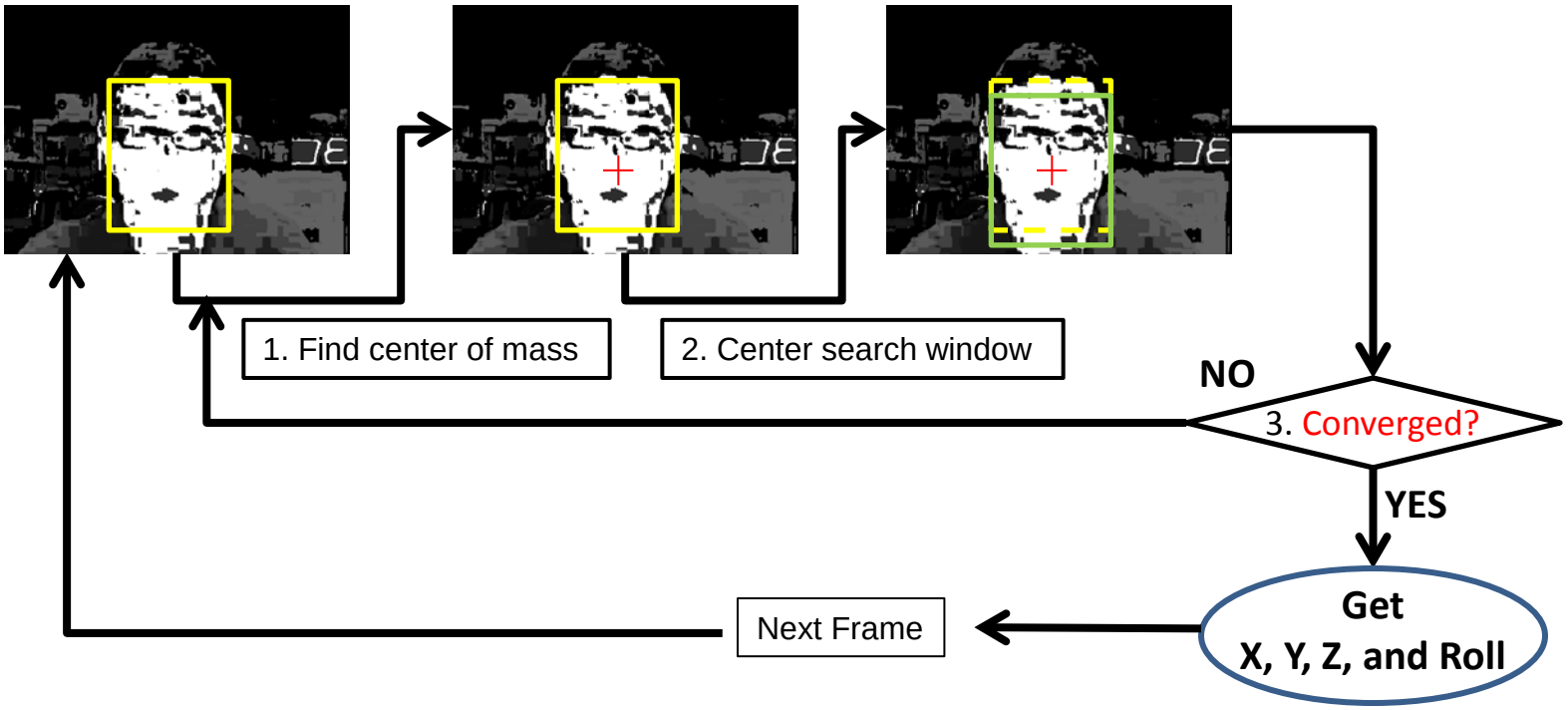
2. System Flowchart: CPD ~~(1/3)~~

- **Color Probability Distribution**
  1. 將 Image 從 RGB 空間轉到 HSV 空間，獲得 Hue 分量。
  2. 根據 Calculation Region 中的 Hue 分量建立Histogram。
  3. 依照Histogram相對應的bin的值，將 Raw Image 轉成 Probability Distribution Image。



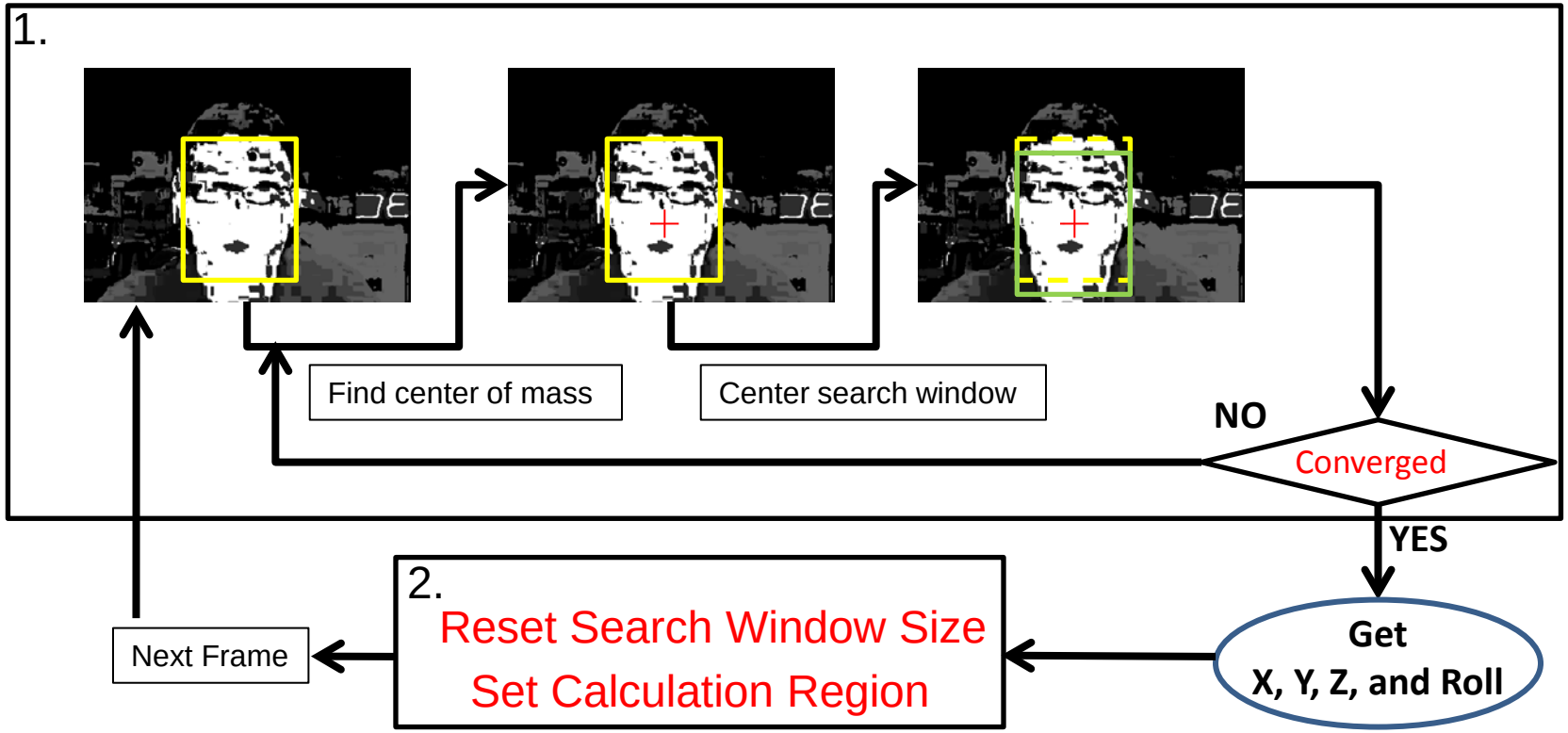
2. System Flowchart: Mean Shift ~~(2/3)~~

- **Mean Shift**
  1. Find center of mass within the search window.
  2. Center search window at the center of mass and find area under it.
  3. Repeat 1. and 2. until convergence (or until the mean location moves less than a preset threshold).
  4. At next frame, update coordinate of center of Search Window.



2. System Flowchart: CAMSHIFT (3/3)

- **CAMSHIFT**
  1. Do Mean Shift as above
  2. At next frame, reset Search Window Size and set Calculation Region and update coordinate of center of Search Window.

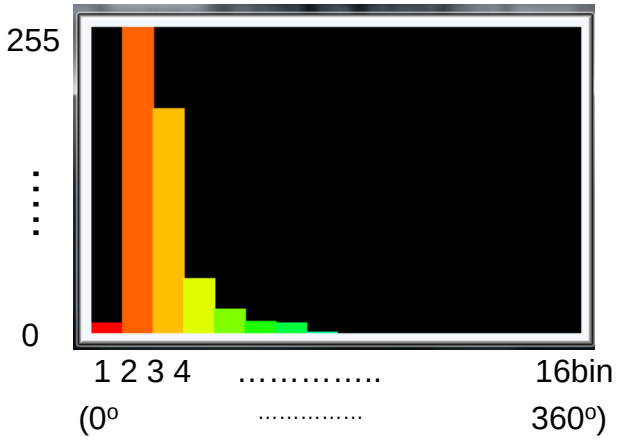


Coupled CAMSHIFT

3. Color Probability Distribution (1/4)

- **Desired Flesh Color**: In order to use CAMSHIFT to track colored objects in a video scene, a probability distribution image of the desired color in the video scene must be created (flesh color in the case of face tracking).
- **HSV Color System:** We use the Hue Saturation Value (HSV) color system to create our color models by taking 1D histograms from the H (Hue) channel in HSV space
- **Histogram:** When sampling is complete, the histogram is saved for future use.

$$\frac{\text{Pixel Number of bin}}{\text{Max Pixel Number of bin}} * 255$$

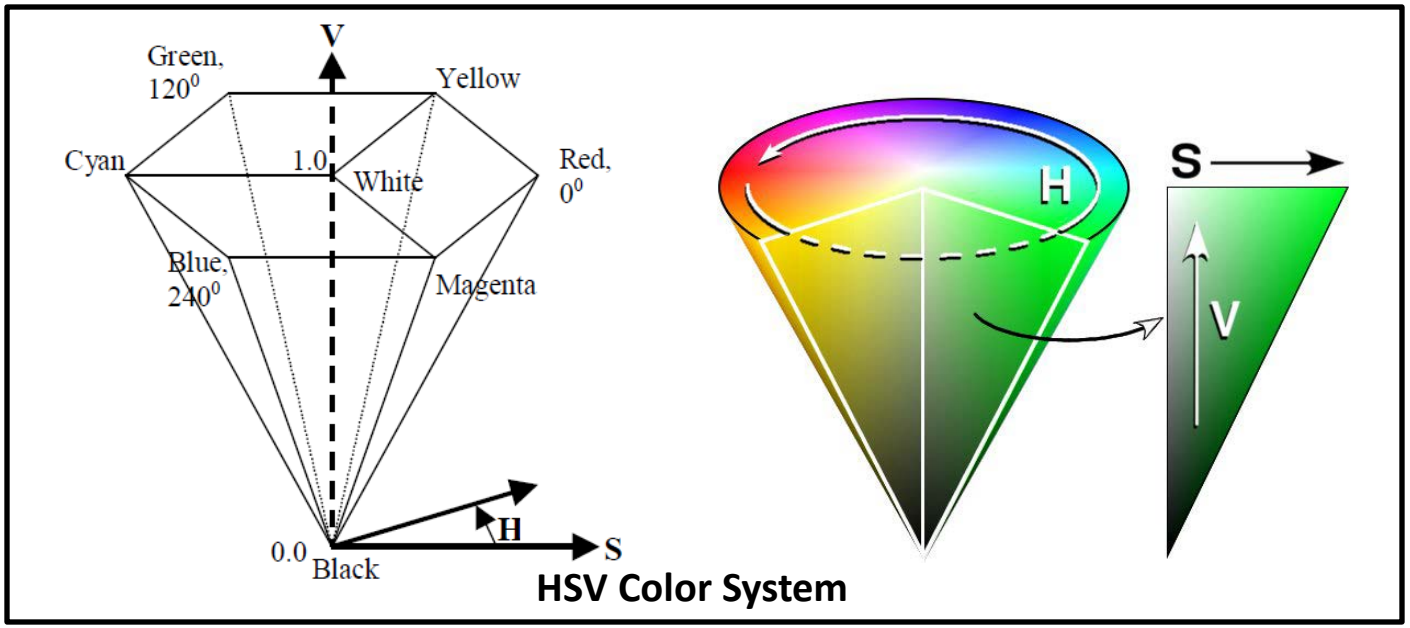


1D Histograms from the Hue channel

3. Color Probability Distribution (2/4)

- Q: What is HSV Color System?
- A: HSV Color System 是 1978 年由 Alvy Ray Smith 創立的，它是經由三原色光模式RGB的非線性變換而來。[From Wikipedia]

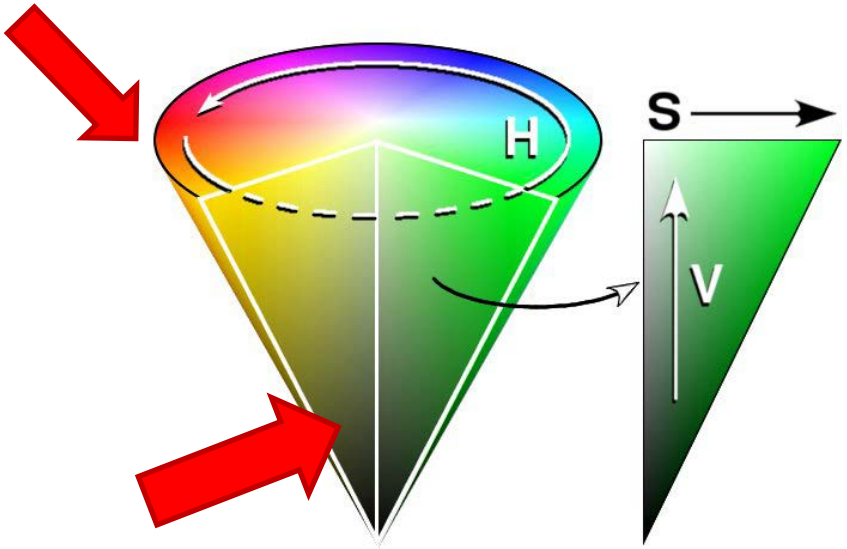
• Hue (色相):	是色彩的基本屬性，就是平常所說的顏色名稱，如紅色、黃色等，取0-360度的數值。（也有用0 –100%的方法確定的）。
• Saturation (飽和度):	是指色彩的純度，越高色彩越純，低則逐漸變灰，取0-100%的數值。
• Value (明度):	也叫「亮度」，取0-100%。



HSV Color System

3. Color Probability Distribution (3/4)

- Q1: Why use Hue from HSV?
- A1:
  - 因為要追蹤人臉，要考慮的是膚色，而Hue是單純的考慮色相，而沒有加入彩度與亮度，這樣一來光線的干擾會較少。
- Q2: 使用 Hue 就不會被 Saturation 與 Value 所干擾?
- A2: NO.
  - When brightness is low (V near 0), saturation is also low (S near 0). Hue then becomes quite noisy.



3. Color Probability Distribution (4/4)

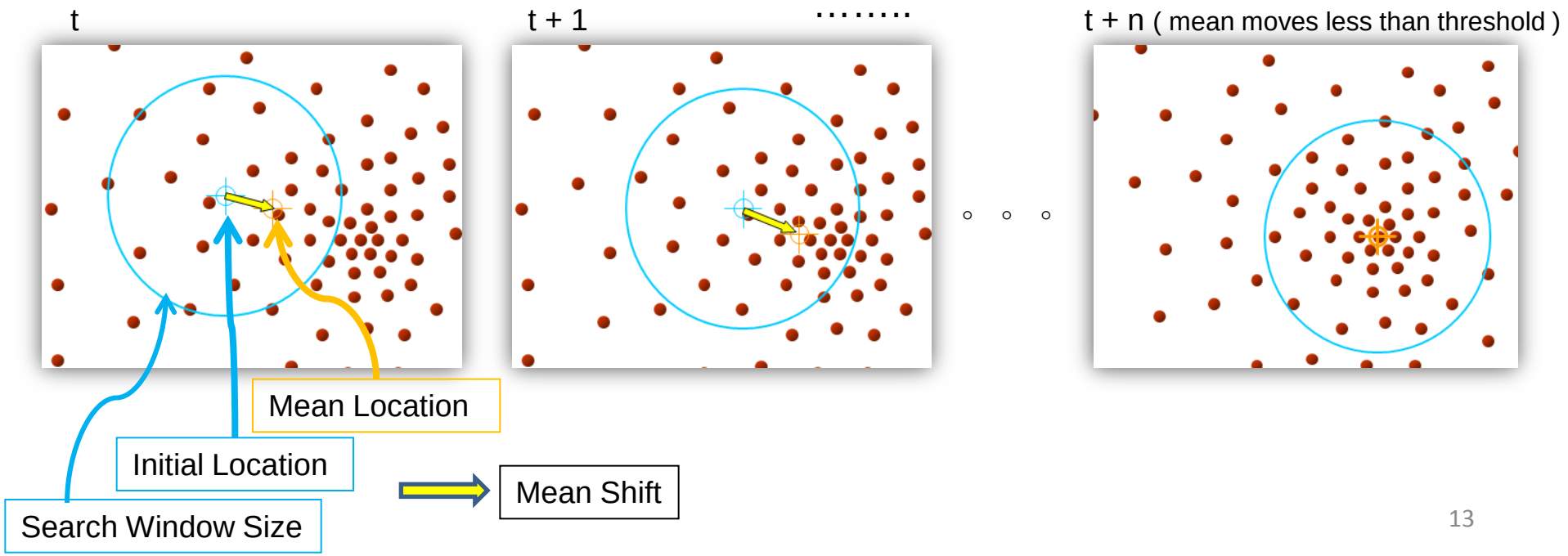
- Q: How to solve above problem?
- A:
  - **Adjust Camera:** For very dim scenes (V near 0), the camera must auto-adjust or be adjusted for more brightness or else it simply cannot track.
  - **Ignore Hue Pixels:** At very low saturation (S near 0), hue is not defined so we also ignore hue pixels that have very low corresponding saturation.

4.Mean Shift (1/4)

- **What is Mean Shift?**
  - An algorithm that iteratively shifts a data point to the average of data points in its neighborhood.
  - Mean Shift can use for clustering, mode seeking, probability density estimation, tracking, etc.

4.Mean Shift (2/4)

- **Procedure of Mean Shift Algorithm**
  1. Choose the initial location of the search window.
  2. Choose a search window size.
  3. Compute the mean location in the search window.
  4. Center the search window at the mean location computed in Step 3.
  5. Repeat Steps 3 and 4 until convergence (or until the mean location moves less than a preset threshold)



4.Mean Shift (3/4)

- **How to Calculate the Mean Shift?**
  - Search Window Size
    - 使用者自訂
  - Initial Location
    - 利用上步驟的 Window Size 的中心點當作 Initial Location.
  - Mean Location
    - Window Size 中資料的中心.

$I(x,y)$: pixel (probability) value at position (x , y) in the image.

Find the zeroth moment $M_{00} = \sum_x \sum_y I(x,y).$

Find the first moment for x and y $M_{10} = \sum_x \sum_y xI(x,y);\ M_{01} = \sum_x \sum_y yI(x,y).$

Then the mean search window location (the centroid) is $x_c = \frac{M_{10}}{M_{00}};\quad y_c = \frac{M_{01}}{M_{00}};$

  - Mean Shift
    - 將 Search Window 的 Initial Location 移向 Mean Location.

4.Mean Shift (4/4)

- Q: Why Mean Shift alone does not work in Tracking?
- A:
  - The fixed-size search window will occur some problem:
    - **Get lost for large object**: Small fixed-sized windows may get lost entirely for large object translation in the scene.
    - **Distract**: Large fixed-sized windows may include distractors (other people or hands).
  - 如何解決?
    - Using the Continuously Adaptive Mean Shift (CAMSHIFT) algorithm.

5.CAMSHIFT (1/8) ??

- **What is CAMSHIFT?**
  - **For Dynamically Changing Distributions:**
    Unlike the Mean Shift algorithm, CAMSHIFT is designed for dynamically changing distributions (???).
  - **Adjust the Search Window Size:**
    The CAMSHIFT algorithm adjusts the search window size in the course of its operation. (Initial window size can be set at any reasonable value)
  - **Use the Zeroth Moment:**
    Instead of a set or externally adapted window size, CAMSHIFT relies on the zeroth moment information, extracted as part of the internal workings of the algorithm, to continuously adapt its window size within or over each video frame.

zeroth moment

$$M_{00} = \sum_x \sum_y I(x, y).$$

5.CAMSHIFT (2/8)

- **Procedure of CAMSHIFT**
  1. Choose the initial location of the search window.
  2. Do Mean Shift as above (one or many iterations); Then store the zeroth moment of image.
  3. Set the search window size equal to a function of the zeroth moment found in Step 2.
  4. Repeat Steps 2 and 3 until convergence (mean location moves less than a preset threshold).

PS:

1. At the same frame: The search window size s does not change to save the computational time.
   Between frames: The search window size will change.
2. Camshift: At the first several frames (for example, steps (=frames) 1st~4th, probably the face region does not equal to search window size s. However, at step or frame 6^{th}, the search window not only reach the face centroid but also extend to the right size of the face.

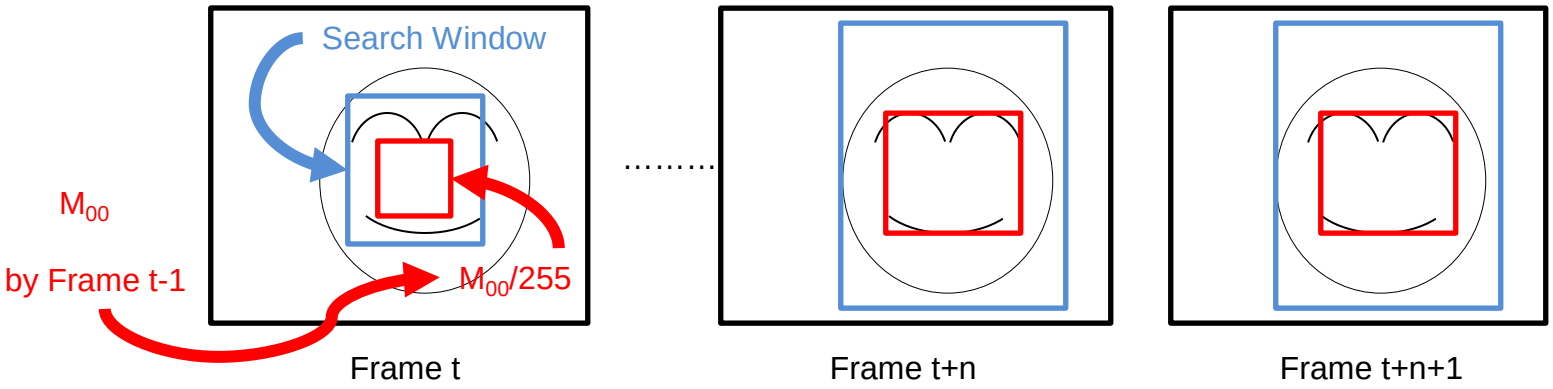
Wu Procedure

5.CAMSHIFT (3/8)

- **How to use zeroth moment to set window size?**
  - For 2D color probability distributions where the maximum pixel value is 255, we set window size s to

$$s = 2 * \sqrt{\frac{M_{00}}{255}}$$

    - To convert the resulting 2D region to a 1D length, we need to take the square root.
    - Our goal is then to track the whole color object so we need an expansive window. Thus, we further multiply the result by two so that the window grows to encompass the connected distribution area.
    - Because face are somewhat elliptical, so we set width to s and length to 1.2s



5.CAMSHIFT (4/8)

- Q1: 為什麼 M_{00} /255 會是 area?
- A1:

如果每個 pixel 的最大值都是255, 假設框住的 region 中, 每個 pixel 皆為最大值, 如下圖所示:

255	255	255
255	255	255
255	255	255

$$M_{00} = \sum_x \sum_y I(x, y).$$

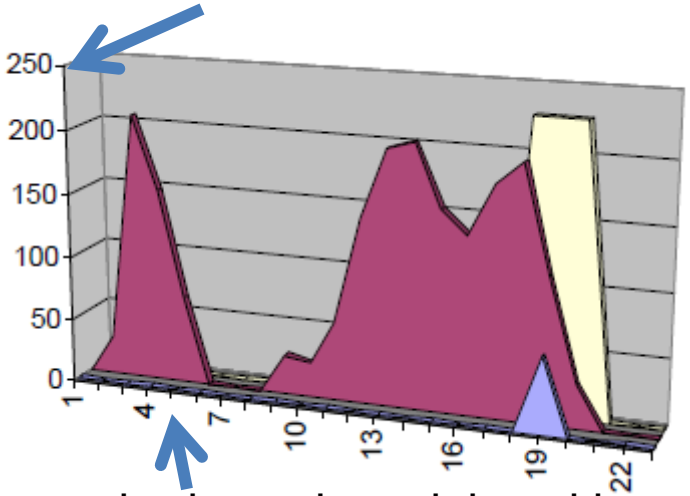
那麼 M_{00} /255 = 9 也就等於是 pixel 的面積, 故在此用這方法來求得面積以及邊長。

If the probability is high, that is, the window size is still smaller, so it multiplies s=2*3=6 to extend the window size.
When the probability is getting lower, that is, the window size already reach the boundary.

5.CAMSHIFT (5/8)

- **Startup of CAMSHIFT**

probability distribution value



horizontal spatial position

1D cross-section of an actual sub-sampled flesh color probability distribution of an image of a face and a nearby hand.

CAMSHIFT search window.

Mean shift point.

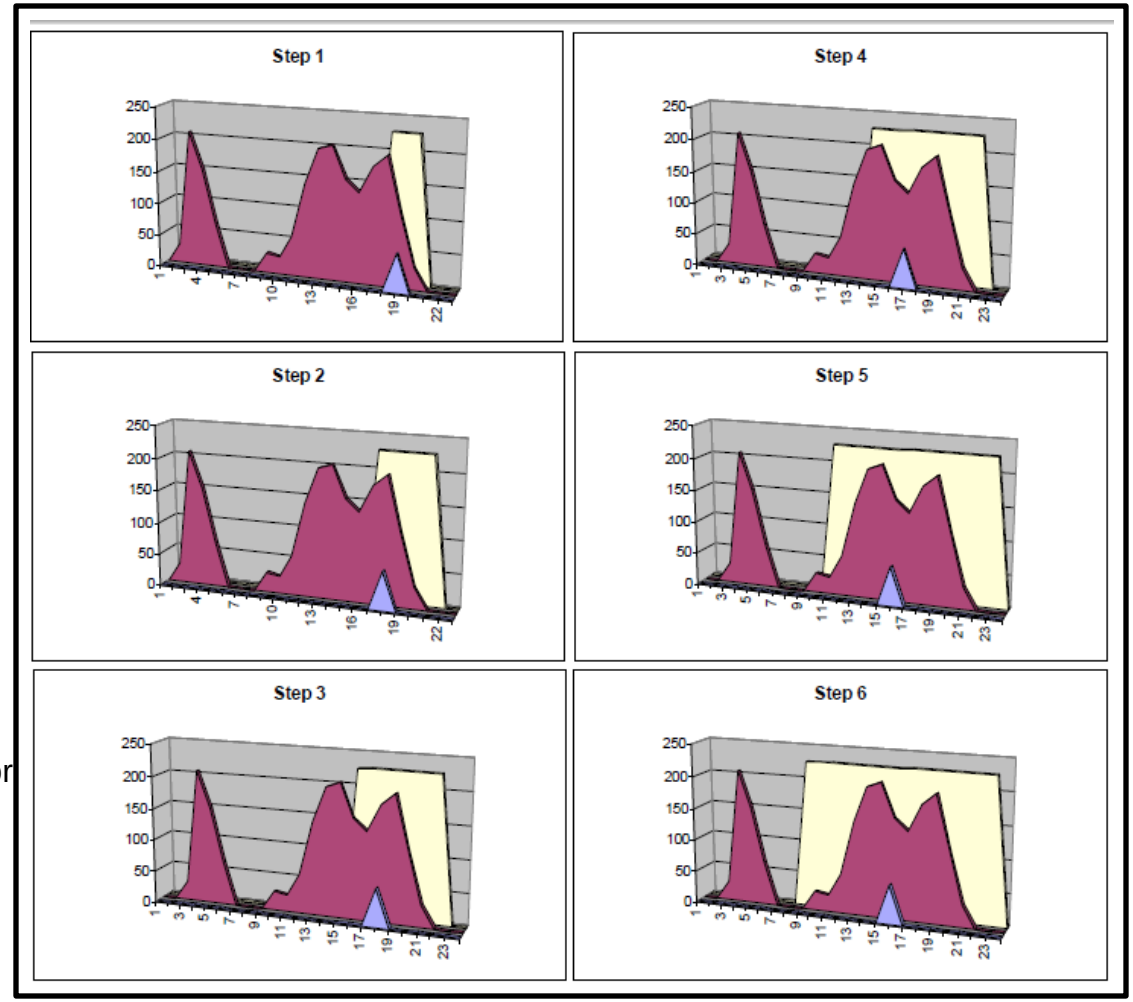
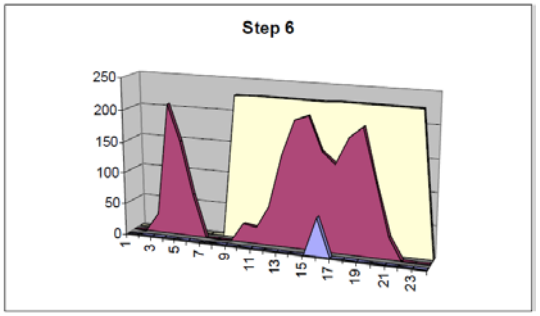


Figure A. Between frames (steps)

5.CAMSHIFT (6/8)

- **Frame to Frame Tracking**
  - In Figure B, the red color probability distribution has shifted left and changed form.
  - At the left in Figure B, the search window starts at its previous location from the bottom right in Figure A.
  - In one iteration it converges to the new face center.



Step 6 of Figure A

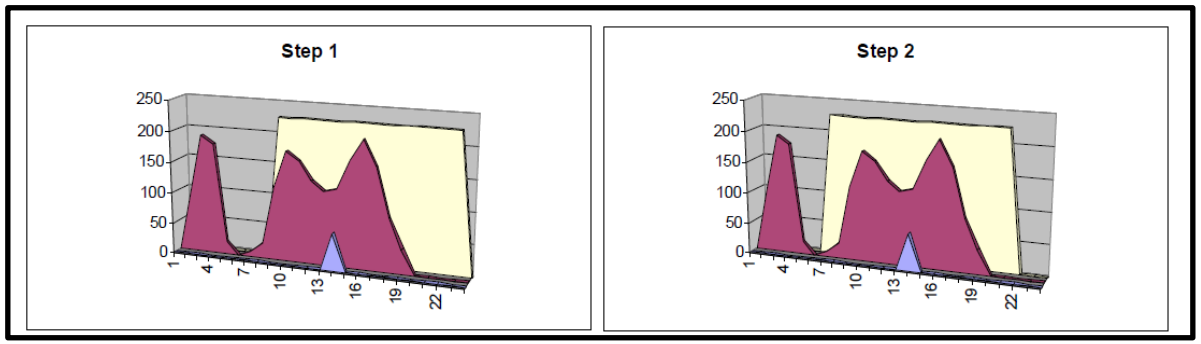


Figure B

5.CAMSHIFT (7/8) ?

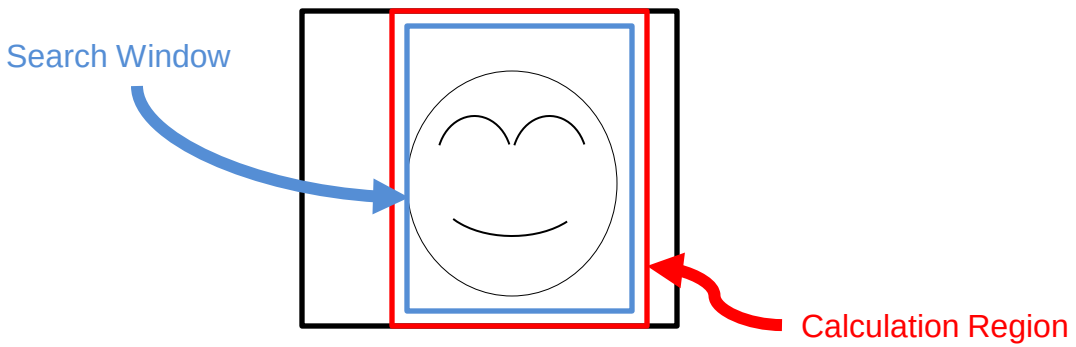
- **Coupled CAMSHIFT Algorithm**
  - When tracking a colored object, CAMSHIFT calculates the centroid of the 2D color probability distribution within its 2D window of calculation, re-centers the window, then calculates the area for the next window size.
  - Thus, we needn't calculate the color probability distribution over the whole image, but can instead restrict the calculation of the distribution to a smaller image region surrounding the current CAMSHIFT window.
  - This tends to result in large computational savings when flesh color does not dominate the image. We refer to this feedback of calculation region size as the Coupled CAMSHIFT Algorithm.

(利用上個Frame的calculation region結果來計算下個Frame的calculation region，降低計算量。)

5.CAMSHIFT (8/8) ?

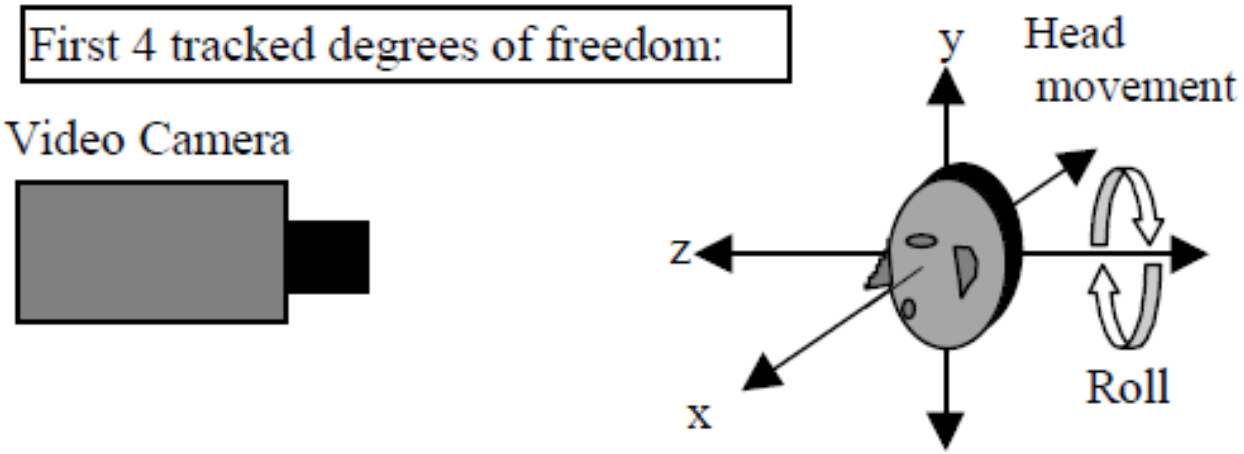
- **Procedure of Coupled CAMSHIFT Algorithm**

1. First, set the calculation region of the probability distribution to the whole image.~~(把整個image當作calculation region~~)
2. Choose the initial location of the 2D mean shift search window.(~~隨意給一個~~initial location，做mean shift運算。)
3. Calculate the color probability distribution in the 2D region centered at the search window location in an area slightly larger than the mean shift window size.(計算機率分布，計算範圍比mean shift的search window大一點點即可。)
4. Mean shift as above. Store the zeroth moment (area or size) and mean location.
5. For the next video frame, center the search window at the mean location stored in Step 4 and set the window size to a function of the zeroth moment found there. Go to Step 3.



5.1 Implementation Detail: xxxx (1/6)

- CAMSHIFT gives us a computationally efficient, simple to implement algorithm that tracks four degrees of freedom. (X, Y, Z, and Roll)



- X: 人臉X方向的移動。
- Y: 人臉Y方向的移動。
- Z: 人臉前後的移動。
- Roll: 人臉的轉動。

5.1 Implementation Detail (2/6)

- **How to obtain 4 degree of freedom X, Y, Z, and Roll?**
- **Calculate X and Y and Z(Distance from the camera/Area) :**
  - Find the zeroth moment

$$M_{00} = \sum_x \sum_y I(x, y).$$

  - Find the first moment for x and y

$$M_{10} = \sum_x \sum_y xI(x, y); \; M_{01} = \sum_x \sum_y yI(x, y).$$

  - Then the mean search window location (the centroid) is

$$x_c = \frac{M_{10}}{M_{00}}; \quad y_c = \frac{M_{01}}{M_{00}};$$

  - **X =** x_c **, Y =** y_c **, Z is proportional to Area** M_{00} So z is not accurate

5.1 Implementation Detail (3/6)

- **How to obtain 4 degree of freedom X, Y, Z, and Roll?**
- **Determine Width and Length of window :**
  - Because face are somewhat elliptical, so we set width to s and length to 1.2s
  - $s = 2 * \sqrt{\frac{M_{00}}{255}}$
  - The maximum distribution value per discrete cell is 255, so we divide the zeroth moment by 255 to convert the calculated area to units of number of cells.
  - To convert the resulting 2D region to a 1D length, we need to take the square root.

5.1 Implementation Detail (4/6)

- **How to track 4 degree of freedom X, Y, Z, and Roll?**
- **Calculate Roll :**
  - Second moments are $$M_{20} = \sum_x \sum_y x^2 I(x,y);\ M_{02} = \sum_x \sum_y y^2 I(x,y)$$
  - The object orientation (major axis) is $$\theta = \frac{\arctan\left(\frac{2\left(\frac{M_{11}}{M_{00}} - x_c y_c\right)}{\left(\frac{M_{20}}{M_{00}} - x_c^2\right) - \left(\frac{M_{02}}{M_{00}} - y_c^2\right)}\right)}{2}$$
  - Let $$a = \frac{M_{20}}{M_{00}} - x_c^2, \quad b = 2\left(\frac{M_{11}}{M_{00}} - x_c y_c\right), \quad c = \frac{M_{02}}{M_{00}} - y_c^2,$$

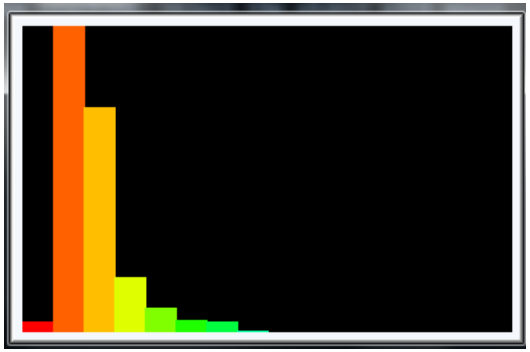
  - Then length l and width w form the distribution centroid are $$l = \sqrt{\frac{(a+c) + \sqrt{b^2 + (a-c)^2}}{2}}, \quad w = \sqrt{\frac{(a+c) - \sqrt{b^2 + (a-c)^2}}{2}}.$$

[4] W.T. Freeman, K. Tanaka, J.Ohta, and K. Kyuma, “Computer Vision for Computer Games,” Int. Conf. On Automatic Face and Gesture Recognition, pp. 100-105, 1996.

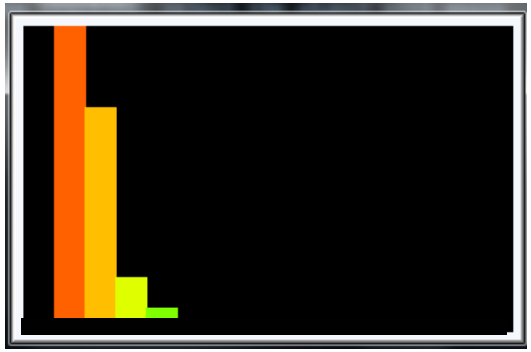
- Why not use only s and 1.2s instead of l and w for face region to play game?
- Ans:
- 1. For game: 用s跟1.2s準確度不足
- 2. For CamShift: 圈人臉時只需圈個大概，and 用s跟1.2s計算量較小。

Noise Reduction and Normalization for Histogram Distribution (5/6)

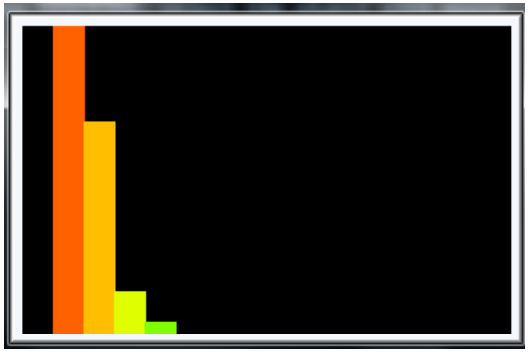
- **Comments on Software Calibration**
  - **Matching Search Window:** Much of CAMSHIFT's robustness to noise, transient occlusions, and distractors depends on the search window matching the size of the object being tracked.
  - **Histogram Adjust:** We adjust the color histogram up or down by a constant, truncating at zero or saturating at the maximum pixel value to control the search window size.
  - For 8-bit hue, we adjust the histogram down by 20 to 80 (out of a maximum of 255), which tends to shrink the CAMSHIFT window to just within the object being tracked and also reduces image noise.



原來的 Histogram



將 5% 砍掉



重新 Normalized

5.1 Implementation Detail (6/6) ?

- **Comments on Hardware Calibration**
  - **Adjust Field of View (FOV)**: The camera's field of view (zoom) must be set so that it covers the space that one intends to track in.
  - **Turn off automatic white balance (WB):** If possible to avoid sudden color shifts.
  - **Adjust other Camera parameter:** Try to set (or auto-adjust)
    - AGC,
    - Shutter speed,
    - iris or CCD integration time so that image brightness is neither too dim nor saturating.

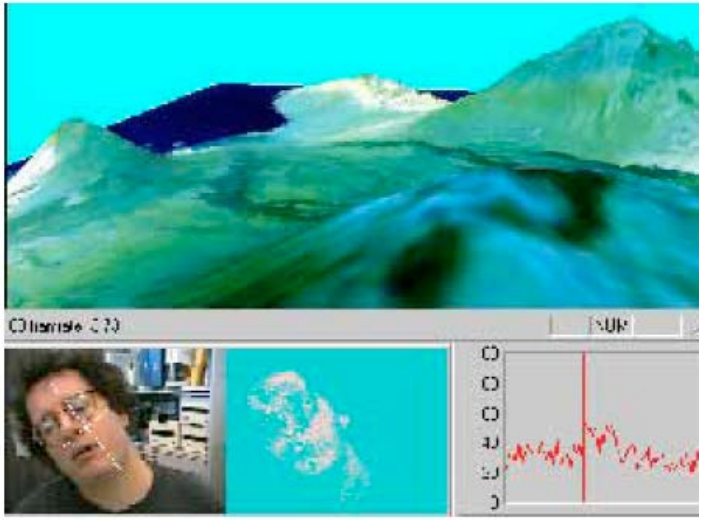
5.2 CAMSHIFT's Use As A Perceptual Interface (1/15)

- **CAMSHIFT's Actual Use as an Interface**
  - By inserting face control variables into the mouse queue, we can control unmodified commercial games such as Quake 2 shown in Figure below.
  - We used:
    1. Left and right head movements to slide a user left and right in the game.
    2. Back and forth head movements to move the user backwards and forwards.
    3. Up or down head movements to let the user shoot (as if ducking or getting jolted by the gun).
    4. Roll left or right to turn the user left or right in the game.

  - This methodology has been used extensively in a series of demos with over 30 different users.

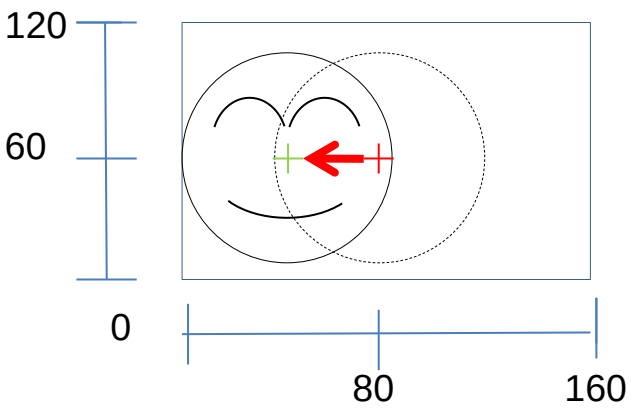
5.2 CAMSHIFT's Use As A Perceptual Interface (2/15)

- **CAMSHIFT's Actual Use as an Interface**
  - Head tracking via CAMSHIFT has also been used to experiment with immersive 3D graphics control in which natural head movements are translated to moving the corresponding 3D graphics camera viewpoint.
  - This has been extensively tested using a 3D graphics model of the Forbidden City in China as well as in exploring a 3D graphics model of the big island of Hawaii as shown in Figure below.



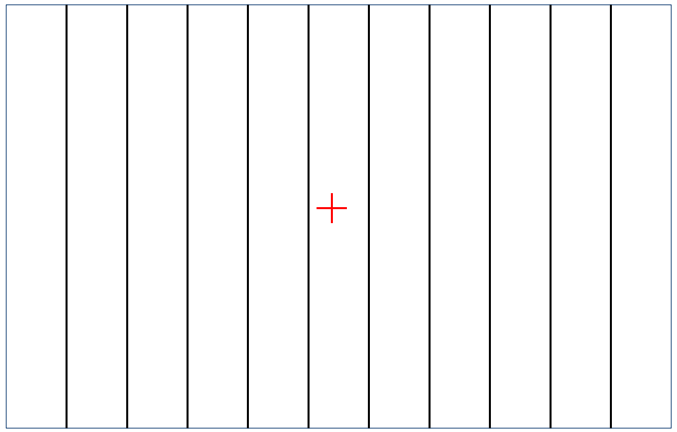
5.2 CAMSHIFT's Use As A Perceptual Interface (3/15)

- **Treatment of CAMSHIFT Tracking Variables for Use in a Perceptual User Interface.**
  - **Neutral position:** For game and graphics control, X, Y, Z, and Roll often require a "neutral" position; that is, a position relative to which further face movement is measured.
  - For example, if the captured video image has dimensions (Y, X) of 120x160, a typical neutral position might be Y=60, X=80. Then if X < 80, the user has moved 80-X left; if X > 80, the user has moved X-80 right and so on for each variable.



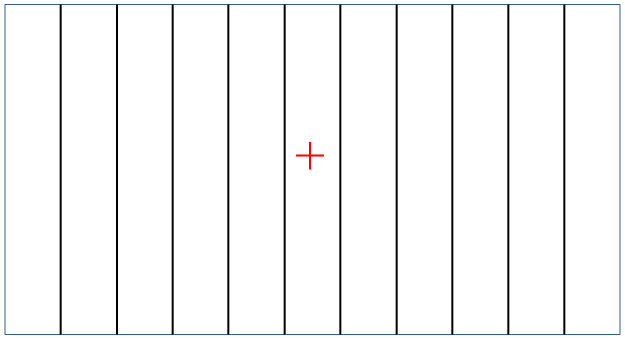
5.2 CAMSHIFT's Use As A Perceptual Interface (4/15)

- **Piecewise Linear Transformation of Control Variables.**
  - To obtain differential control at various positions including a jitter-damping neutral movement region, each variable's relative movement (above or below the neutral position) is scaled in "N" different ranges.
  - In the X variable example above, if X is in range #1, X would be scaled by "X scale 1"; if X is in range #2, X would be scaled by "X scale 2" and so on.



5.2 CAMSHIFT's Use As A Perceptual Interface (5/15)

- Q: Why use Piecewise Linear Transformation of Control Variables?
- A:
  - 在遊戲時，頭移動的幅度 (離neutral的距離) 越大，通常就是想要遊戲畫面移動的距離越大，如果每個X的移動，都是相同的距離，會有跟不上的感覺。
  - 可以將 X 與 neutral 距離分成 N 個 range，每個 range 有不同的 scale，距離越大的 range，scale 就相對的越大。



…b2 b1 b1 b2….

5.2 CAMSHIFT's Use As A Perceptual Interface (6/15)

- **The formula for mapping captured video head position P to a screen movement factor F.**

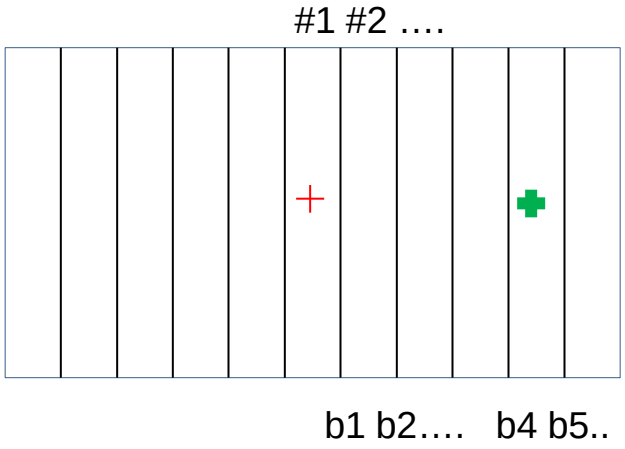
$$F = \min(b_1, P)s_1 + [\min(b_2 - b_1, P - b_1)]^+ s_2 + \ldots + [\min(b_{(i+1)} - b_i, P - b_i)]^+ s_{(i+1)} + \ldots + [P - b_{(N-1)}]^+ s_N$$ **(control equation 1)**

  - Where $[\#]^+ \begin{cases} \# & \#>0 \\ 0 & \text{otherwise} \end{cases}$
  - The min(A, B) returns the minimum of A or B.
  - Factor $b_1 \rightarrow b_N$ represents the bounds of the ranges.
  - Factor $s_1 \rightarrow s_N$ are the corresponding scale factors for each range.
  - Factor P is the absolute value of the difference of the variable's location from neutral.

5.2 CAMSHIFT's Use As A Perceptual Interface (7/15)

- **The formula for mapping captured video head position P to a screen movement factor F.**

$$F = \min(b_1, P)s_1 + [\min(b_2 - b_1, P - b_1)]^+ s_2 + \ldots + [\min(b_{(i+1)} - b_i, P - b_i)]^+ s_{(i+1)} + \ldots + [P - b_{(N-1)}]^+ s_N$$ **(control equation 1)**



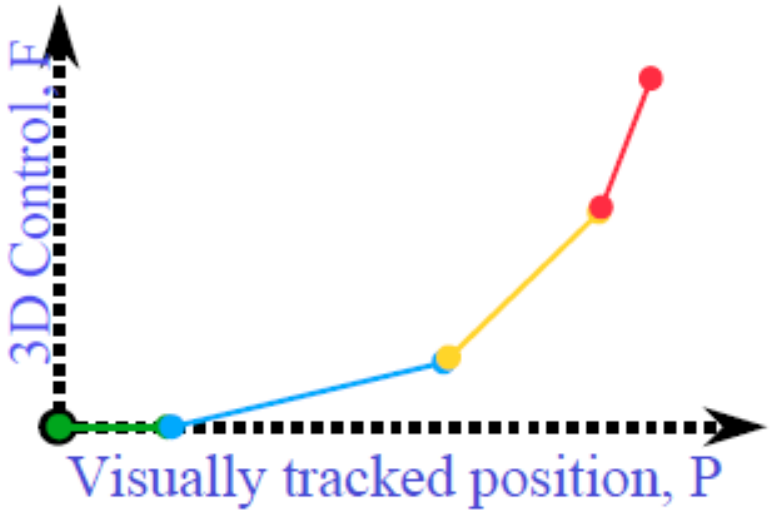
$$F = \min(b_1, P)s_1 + [\min(b_2 - b_1, P - b_1)]^+ s_2 + [\min(b_3 - b_2, P - b_2)]^+ s_3 + [\min(b_4 - b_3, P - b_3)]^+ s_4 + [\min(b_5 - b_4, P - b_4)]^+ s_5 + [\min(b_6 - b_5, P - b_5)]^+ s_6 + \begin{cases} 0 \\ \vdots \\ + [P - b_{(N-1)}]^+ s_N \end{cases}$$

5.2 CAMSHIFT's Use As A Perceptual Interface (8/15)

- **The formula for mapping captured video head position P to a screen movement factor F.**

$$F = \min(b_1, P)s_1 + [\min(b_2 - b_1, P - b_1)]^+ s_2 + \ldots + [\min(b_{(i+1)} - b_i, P - b_i)]^+ s_{(i+1)} + \ldots + [P - b_{(N-1)}]^+ s_N$$ **(control equation 1)**

  - This allows for stability close to the neutral position, with growing acceleration of movement as the distance away from neutral increases, in a piecewise linear manner as shown in this figure .



5.2 CAMSHIFT's Use As A Perceptual Interface (9/15)

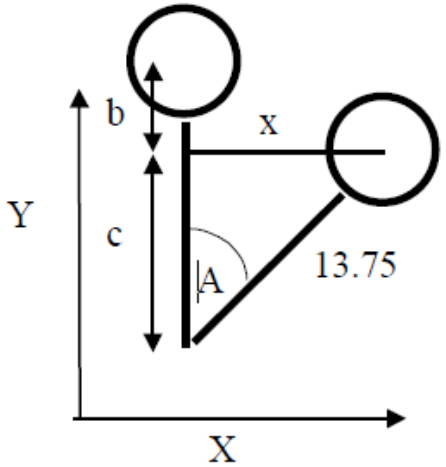
- **Frame Rate Adjustment**
  - If the graphic's rendering rate R can be determined, the final screen movement S is:

    $S = F/R$ **(control equation 2)**

  - Computer graphics and game movement commands S can be issued on each rendered graphic's frame. Thus, it is best for the movement amount S to be sensitive to frame rate.

5.2 CAMSHIFT's Use As A Perceptual Interface (10/15)

- **Y Variable Special Case for Seated User**
  - If a user sits facing the camera (neutral X,Y) and then leans left or right (X movement) by pivoting on his/her chair, Y will decrease as shown in Figure below.
  - **Face Size Proportional to Body Size:** In order to overcome this, we use an empirical observation that the 2nd Eigenvalue (face half width) of the local face flesh color distribution length is proportional to face size, which is, on average, often proportional to body size.



5.2 CAMSHIFT's Use As A Perceptual Interface (11/15)

- Q: Why should Y distance be corrected?
- A:
  - 使用者坐在椅子上時，可能會些微的左右擺動，此時，使用者並沒有想要對遊戲下達指令，但是些微擺動會造成Y方向的移動，所以我們必須修正Y。

5.2 CAMSHIFT's Use As A Perceptual Interface (12/15)

- **Y Variable Special Case for Seated User**
  - Empirically, the ratio of the 2nd Eigenvector to torso length from face centroid is:

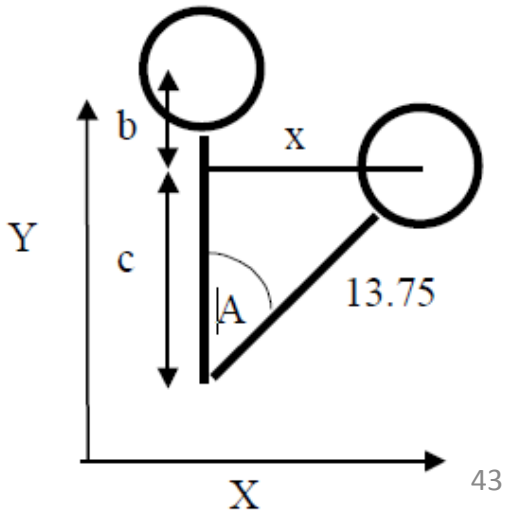
    1 to 13.75 (2 inches to 27.5 inches).**(control equation 3)**

  - Given lean distance x (in 2nd Eigenvector units), and seated size of 13.75, as in Figure below so that sin(A) = x/13.75. Then,

    $A = \sin^{-1}(x/13.75)$, **(control equation 4)**

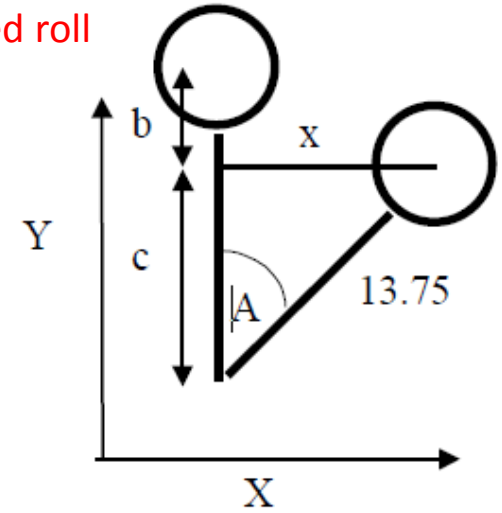
    $b = 13.75(1 - \cos(A))$ **(control equation 5)**

  - This is the Y distance to correct for (add back) when leaning.



5.2 CAMSHIFT's Use As A Perceptual Interface (13/15)

- **Roll Considerations**
  - As can be seen from Figure below, for seated users lean also induces a change in head roll by the angle A. Thus, for control that relies on head roll, this lean-induced roll should be corrected for.
  - Correction can be accomplished in two ways:
    - **Adjust Range Boundary & Scale:** Make the first range boundary b1 in control equation 1 large enough to contain the changes in face orientation that result from leaning. Then use a scale value s1 = 0 so that leaning causes no roll.
    - **Compare with angle A (Eq4):** Subtract the measured roll from the lean-induced roll, A, calculated in control Equation 4 above.



5.2 CAMSHIFT's Use As A Perceptual Interface (14/15)

- **Roll Considerations-when user looking down**
  - Looking down too much causes the forehead to dominate the view which in turn causes the face flesh color "blob" to look like it is oriented horizontally.
  - **Roll Quality:** To correct for such problems, we define a new variable, Q called "Roll Quality." Q is the ratio of the first two Eigenvalues, length l and width w, of the distribution color "blob" in the CAMSHIFT search window:

$$Q = l/w. \qquad \textbf{(control equation 6)}$$

5.2 CAMSHIFT's Use As A Perceptual Interface (15/15)

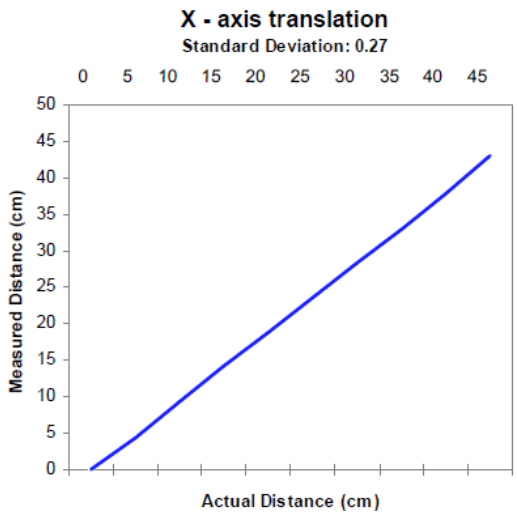
- **Roll Considerations**
  - For problem views of the face such as in Figure below, we observe that Roll Quality is nearly 1.0.
  - For face tracking, roll should be ignored (treated as vertical) for quality measures less than 1.25.
  - Roll should also be ignored for very high quality scores greater than 2.0 since such distributions are un-facelike and likely to have resulted from noise or occlusions

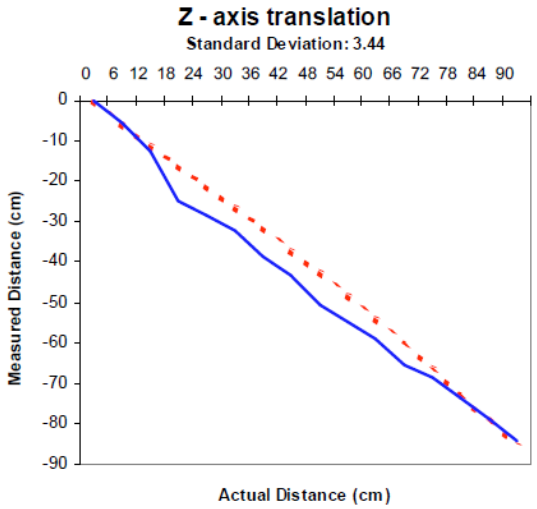
6.CAMSHIFT Analysis (1/14)

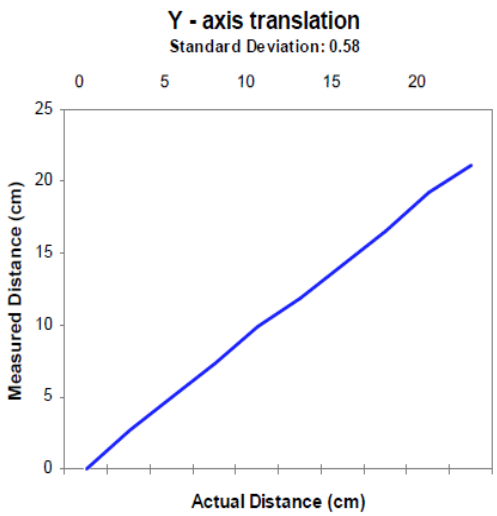
- **Comparison to Polhemus (1/3)**
  - In order to assess the tracking accuracy of CAMSHIFT, we compared its accuracy against a Polhemus tracker.
  - The observed accuracy of Polhemus is +/- 1.5cm in spatial location and about 2.5^{o} in orientation within 30 inches of the Polhemus antenna.
  - We compared Polhemus tracking to CAMSHIFT color object tracking using a 320x240 image size.

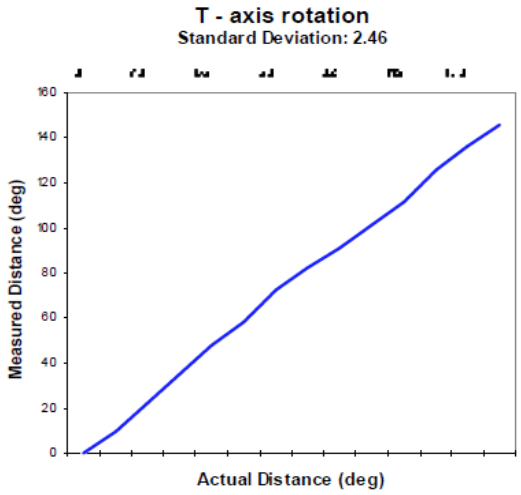
6.CAMSHIFT Analysis (2/14)

- **Comparison to Polhemus (2/3)**
  - Accuracy comparison of Polhemus and CAMSHIFT.









6.CAMSHIFT Analysis (3/14)

- **Comparison to Polhemus (3/3)**
  - The comparison between CAMSHIFT and Polhemus in each of X, Y, Z, and Roll yielded the results shown Table 1.

Tracking Variable	X	Y	Z	Roll
Standard Deviation of Difference	0.27 cm	0.58 cm	3.4 cm	2.4 ^{o}

Table 1

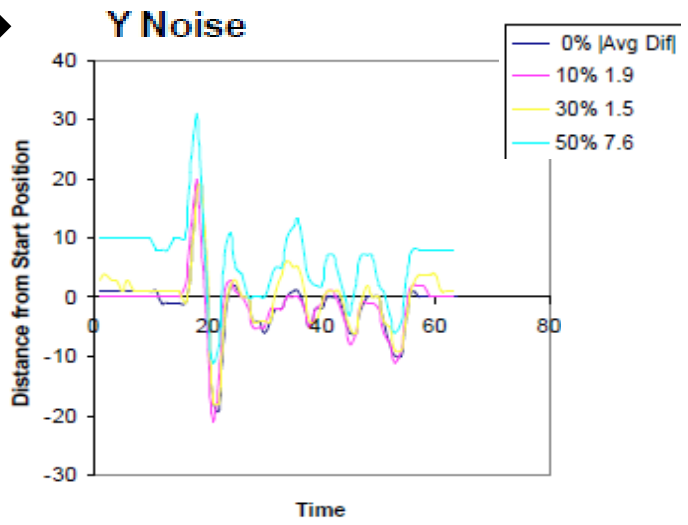
  - Z exhibited the worst difference because CAMSHIFT determines Z by measuring color area, which is inherently noisy.

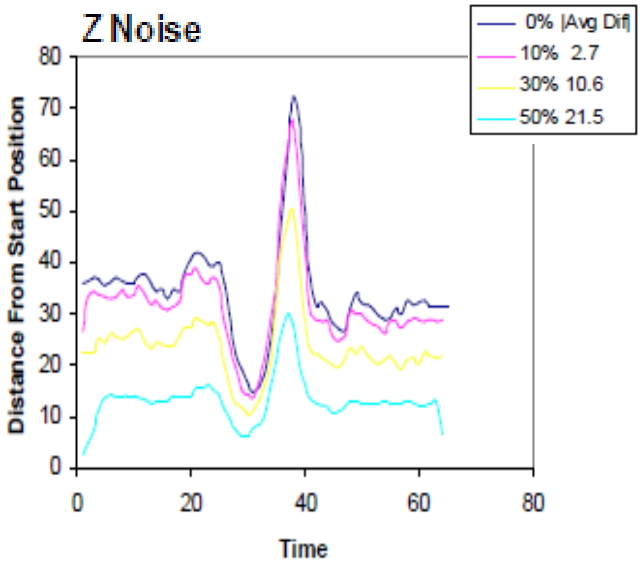
6.CAMSHIFT Analysis (4/14)

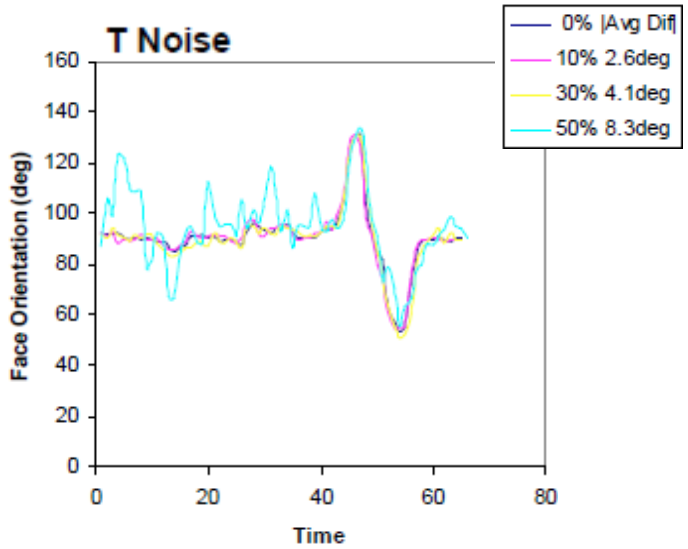
- **Tracking in Noise (1/3)**
  - CAMSHIFT's robust ability to find and track the mode of a dynamically changing probability distribution also gives it good tracking behavior in noise.
  - We videotaped a head movement sequence and then played it back adding 0%, 10%, 30%, and 50% uniform noise.

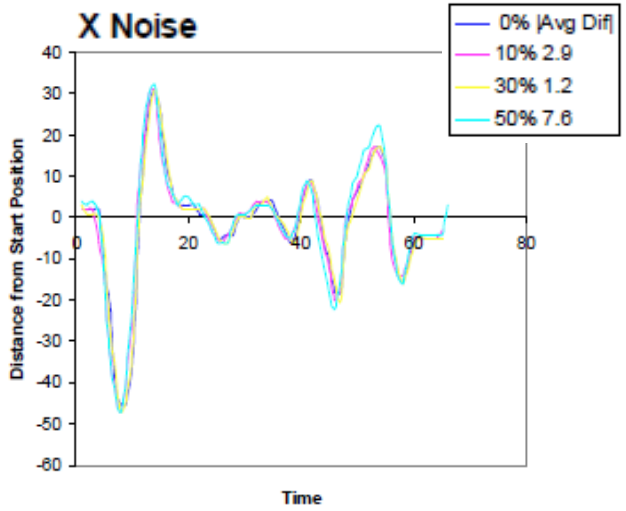
Image of 50% uniform noise

6.CAMSHIFT Analysis (5/14)

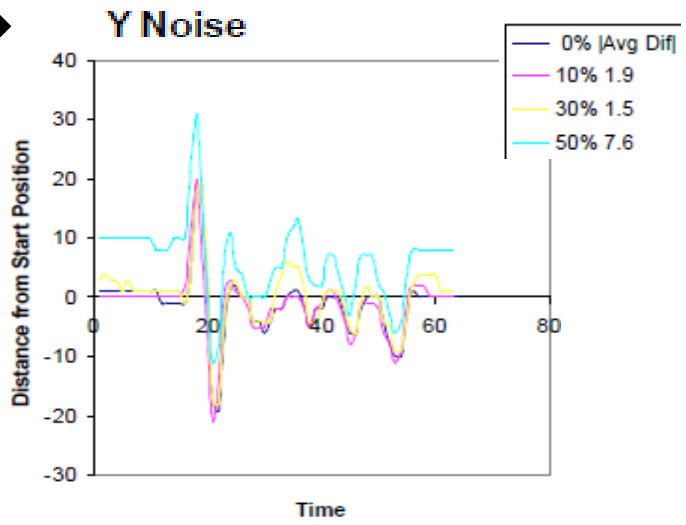
- **Tracking in Noise (2/3)**
  - 

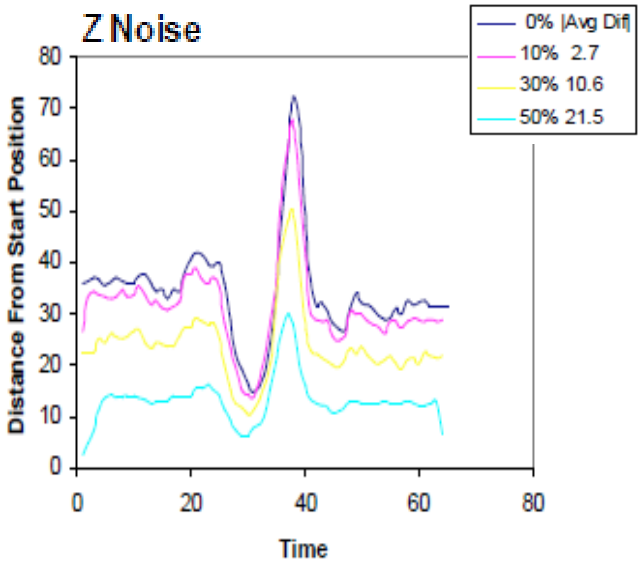


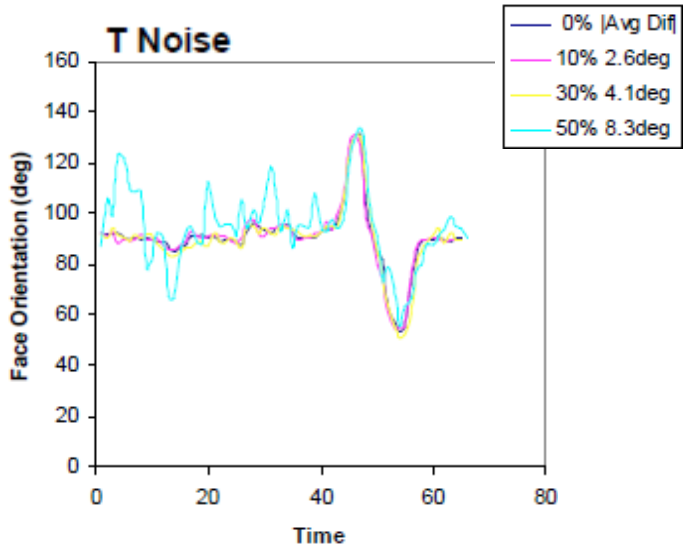




6.CAMSHIFT Analysis (6/14)

- **Tracking in Noise (3/3)**
  - 





Y shows an upward shift simply because the narrower chin(下巴) region exhibits more degradation in noise than the wider forehead.

Z is more of a problem because CAMSHIFT measures Z by tracking distribution area under its search window, and that area is highly effected by noise.

Roll tracks well until noise is such that the length and width of the face color distribution are obscured.

6.CAMSHIFT Analysis (7/14)

- **Tracking in the Presence of Distractions (1/4)**
  - In such cases, CAMSHIFT is robust against distracting (nearby) distributions and transient occlusions, because the search window rarely contains the distractor as shown.
  - Table 2 shows the results collected from 44 sample point on five tracking runs with active background face distraction such as that shown.

Tracked Variable	Average Std. Deviation	Maximum Std. Deviation
X (pixels)	0.42	2.00
Y(pixels)	0.53	1.79
Z(pixels)	0.54	1.50
Roll (degrees)	5.18	46.80

Table 2

6.CAMSHIFT Analysis (8/14)

- **Tracking in the Presence of Distractions (2/4)**
  - Roll is more strongly affected since even a small intersection of a distractor in CAMSHIFT's search window can change the effective orientation of the flesh pixels as measured by CAMSHIFT.

Tracked Variable	Average Std. Deviation	Maximum Std. Deviation
X (pixels)	0.42	2.00
Y(pixels)	0.53	1.79
Z(pixels)	0.54	1.50
Roll (degrees)	5.18	46.80

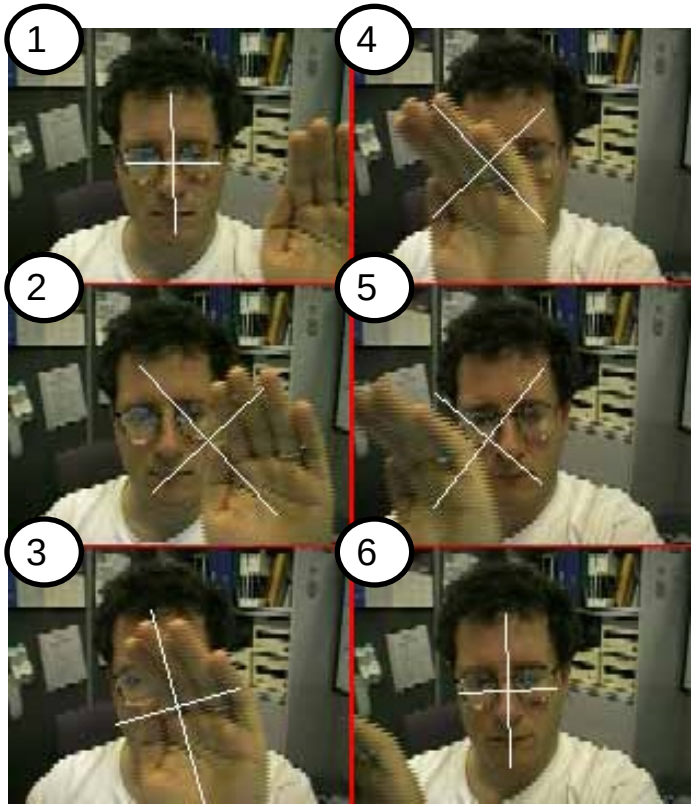
Table 2

6.CAMSHIFT Analysis (9/14)

- **Tracking in the Presence of Distractions (3/4)**
  - CAMSHIFT tends to be robust against transient occlusion because the search window will tend to first absorb the occlusion and then stick with the dominant distribution mode when the occlusion passes.
  - Table 3 shows the results collected from 43 sample points on five tracking runs with active transient hand occlusion of the face.

Tracked Variable	Average Std. Deviation	Maximum Std. Deviation
X (pixels)	2.35	7.17
Y(pixels)	2.81	6.29
Z(pixels)	2.10	4.65
Roll (degrees)	14.64	34.40

Table 3



6.CAMSHIFT Analysis (10/14)

- **Tracking in the Presence of Distractions (4/4)**
  - Roll is more strongly effected due to the arbitrary orientation of the hand as it passes through the search window.

Tracked Variable	Average Std. Deviation	Maximum Std. Deviation
X (pixels)	2.35	7.17
Y(pixels)	2.81	6.29
Z(pixels)	2.10	4.65
Roll (degrees)	14.64	34.40

Table 3

6.CAMSHIFT Analysis (11/14)

- **Performance**
  - The order of complexity of CAMSHIFT is $O(aN^2)$ where a is some constant, and the image is taken to be N x N.
  - Factor a is most influenced by the moment calculations and the average number of mean shift iterations until convergence.
  - The biggest computational savings come through scaling the region of calculation to an area around the search window size as previously discussed.

6.CAMSHIFT Analysis (12/14)

- **Performance**
  - CAMSHIFT was run on a 300 MHz Pentium® II processor, using an image size of 160x120 at 30 frames per second (see Figure C).
  - In Figure C when the tracked face is far from the camera, the CAMSHIFT thread consumes only 10% of the CPU cycles. When the face fills the frame, CAMSHIFT consumes 55% of the CPU.

Figure C

6.CAMSHIFT Analysis (13/14)

- **Performance**
  - Figure D traces computer vision thread's performance in an actual control task of "flying" over a 3D model of Hawaii using head movements.
  - In this case, the average CPU usage was 29%.
  - VTUNE™ analysis showed that the actual CAMSHIFT operation (excluding image capture, color conversion or image copying) consumed under 12% of the CPU.

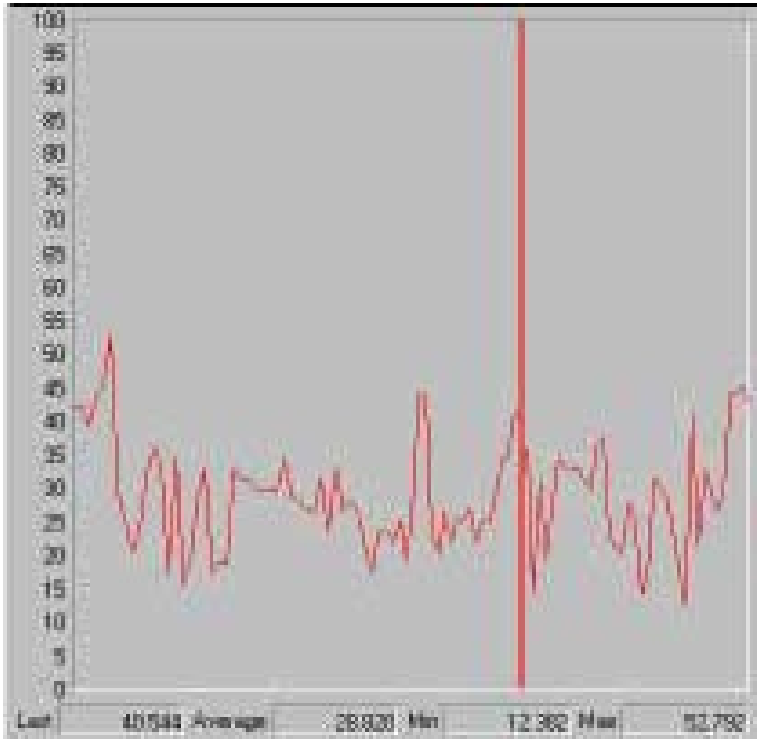


Figure D

6.CAMSHIFT Analysis (14/14)

- **Performance**
  - The MMX™ technology optimized image moments calculation has recently been added to the Image Processing Library, but not in time for publication. The use of such optimized moment calculations will boost performance noticeably since this forms part of the inner mean shift calculation loop.
  - Even without this improvement, CAMSHIFT's current average actual use efficiency of 29% allows it to be used as a visual user interface.

7.Discussion

- We've seen that CAMSHIFT handles these problems as follows:
  - **Irregular object motion**: CAMSHIFT scales its search window to object size thus naturally handling motion irregularities.
  - **Image noise**: The color model eliminates much of the noise, and CAMSHIFT tends to ignore the remaining outliers.
  - **Distractors**: CAMSHIFT ignores objects outside its search window so objects such as nearby faces and hands do not affect CAMSHIFT's tracking.
  - **Occlusion**: As long as occlusion isn't 100%, CAMSHIFT will still tend to follow what is left of the objects' probability distribution.
  - **Lighting variation**: Using only hue from the HSV color space and ignoring pixels with high or low brightness gives CAMSHIFT wide lighting tolerance.
- **Limitations:**
  - **Z 容易受雜訊影響:** Since CAMSHIFT derives Z from object area estimates, Z is subject to noise and spurious values.
  - **容易被色彩訊息影響:** CAMSHIFT relies on color distributions alone, errors in color (colored lighting, dim illumination, too much illumination) will cause errors in tracking.

8.Conclusion

- CAMSHIFT is a simple, computationally efficient face and colored object tracker.
- While acknowledging the limitation imposed by its simplicity, we can still see that CAMSHIFT tracks virtually as well as more expensive tethered trackers (Polhemus) or much more sophisticated, computationally expensive vision systems [17].
- In this project, we designed a highly efficient face tracking algorithm rather than a more complex, higher MIPs usage algorithm.
- In this way, the functionality of the computer vision interface will increase with increasing Intel CPU speeds. A user will thus be able to upgrade their computer vision interface by upgrading to higher speed Intel CPUs in the future.

[17]A. Azabayejani, T. Starner, B. Horowitz, and A. Pentland, “Visually Controlled Graphics,” IEEE Tran. Pat. Anal. and Mach. Intel. pp. 602-605, Vol. 15, No. 6, June 1993.

9. Reference

- [4] W.T. Freeman, K. Tanaka, J.Ohta, and K. Kyuma, “Computer Vision for Computer Games,” Int. Conf. On Automatic Face and Gesture Recognition, pp. 100-105, 1996.
- [17] A. Azabayejani, T. Starner, B. Horowitz, and A. Pentland, “Visually Controlled Graphics,” IEEE Tran. Pat. Anal. and Mach. Intel. pp. 602-605, Vol. 15, No. 6, June 1993.
- [18] Y. Cheng, “Mean shift, mode seeking, and clustering,” IEEE Trans. Pattern Anal. Machine Intell., 17:790-799, 1995.

Thank you