Computer Vision Face Tracking For Use in a Perceptual User Interface

用于感知用户界面的计算机视觉人脸跟踪

Gary R. Bradski, Microcomputer Research Lab, Santa Clara, CA, Intel Corporation

Index words: computer vision, face tracking, mean shift algorithm, perceptual user interface, 3D graphics interface
索引：计算机视觉，人脸追踪，mean shift 算法，感知用户界面，3D 图像界面

Abstract

As a first step towards a perceptual user interface, a computer vision color tracking algorithm is developed and applied towards tracking human faces. Computer vision algorithms that are intended to form part of a perceptual user interface must be fast and efficient. They must be able to track in real time yet not absorb a major share of computational resources: other tasks must be able to run while the visual interface is being used. The new algorithm developed here is based on a robust non-parametric technique for climbing density gradients to find the mode (peak) of probability distributions called the mean shift algorithm. In our case, we want to find the mode of a color distribution within a video scene. Therefore, the mean shift algorithm is modified to deal with dynamically changing color probability distributions derived from video frame sequences. The modified algorithm is called the Continuously Adaptive Mean Shift (CAMSHIFT) algorithm. CAMSHIFT's tracking accuracy is compared against a Polhemus tracker. Tolerance to noise, distractors and performance is studied.

CAMSHIFT is then used as a computer interface for controlling commercial computer games and for exploring immersive 3D graphic worlds.

摘要

作为搭建感知用户界面的第一步，一种计算机视觉颜色追踪算法被开发出来并应用于人脸追踪。计算机视觉算法的目的是组成部分感知用户界面，因此算法必须快速高效。这些算法必须能够实现实时追踪但不能占用大量计算机资源，这意味着其他的任务在可视化界面被使用也必须能够运行。此处开发的新算法是基于一个名为 mean shift 算法的具有鲁棒性的非参数算法，该算法对密度通过梯度上升法得到概率分布的模式（峰值）。在我们的应用场合下，我们想得到一段视频中的颜色分布模式。因此，我们修改了 mean shift 算法，以处理来自视频帧序列的动态变化的颜色概率分布。这种修改的算法被称作连续自适应的 mean shift 算法（CAMSHIFT）。我们就追踪精度对 CAMSHIFT 和 Polhemus 公司的追踪设备进行比较，并且研究了 CAMSHIFT 算法对于噪声干扰的容忍性。

CAMSHIFT 被用作控制商业电脑游戏和探索浸入式的三维图形世界的计算机接口。

Introduction

This paper is part of a program to develop a Perceptual User Interface for computers. Perceptual interfaces are ones in which the computer is given the ability to sense and produce analogs of the human senses, such as allowing computers to perceive and produce localized sound and speech, giving computers a sense of touch and force feedback, and in our case, giving computers an ability to see. The work described in this paper is part of a larger effort aimed at giving computers the ability to segment, track, and understand the pose, gestures, and emotional expressions of humans and the tools they might be using in front of a computer or settop box. In this paper we describe the development of the first core module in this effort: a 4-degree of freedom color object tracker and its application to flesh-tone-based face tracking. Computer vision face tracking is an active and developing field, yet the face trackers that have been developed are not sufficient for our needs. Elaborate methods such as tracking contours with snakes [[10][12][13]], using Eigenspace matching techniques [14], maintaining large sets of statistical hypotheses [15], or convolving images with feature detectors [16] are far too computationally expensive. We want a tracker that will track a given face in the presence of noise, other faces, and hand movements. Moreover, it must run fast and efficiently so that objects may be tracked in real time (30 frames per second) while consuming as few system resources as possible. In other words, this tracker should be able to serve as part of a user interface that is in turn part of the computational tasks that a computer might routinely be expected to carry out. This tracker also needs to run on inexpensive consumer cameras and not require calibrated lenses.

简介

这篇论文是开发计算机感知用户界面项目的一部分。感知界面是指赋予计算机能够感知并产生和人类感觉类似的能力，比如允许计算机感知和产生本地化的声音和语音，给计算机触觉和力量的反馈，在我们的应用场合中则是赋予计算机“看”的能力。本文所描述的工作是一个更大的目标的一部分，这个更大的目标旨在赋予计

算机以及可能用在计算机前端或机顶盒上的工具分割，跟踪，理解姿势，手势和人类情感表达的能力。本文中我们描述的是这个更大的目标中第一个核心模块，一个 4 自由度的彩色目标跟踪和基于肤色的人脸跟踪中的应用。

使用计算机视觉进行脸部追踪是一个活跃并在发展中的研究领域，然而目前已开发的脸部追踪技术并不能满足我们的需求。一些精细的方法比如结合 snakes 算法进行轮廓追踪，使用特征匹配技术，进行大量统计学假设，或进行卷积图像特征检测都使用了太多的计算资源。我们想要一个能够对一个有噪声影响的给定的脸部、其他脸部、手的移动进行追踪的追踪器，此外它必须快速，高效地运行，以便可以实时跟踪对象（每秒 30 帧），并尽量少占用系统资源。换句话说，这个跟踪器应该能够作为一个计算机任务中的用户界面的一部分，这个计算机任务可能会经常被切换。这个跟踪器还应能运行在廉价的消费级相机上并且使用时不需要校准镜头。因此，为了找到一种快速简单的基本跟踪算法，我们一直专注于研究基于颜色的追踪，但即使是这些简单的算法由于使用了颜色相关性检测、斑点检测、区域增长法、卡尔曼滤波预测、轮廓匹配，都有复杂的计算过程（因此在任何给定的 CPU 速度下都较慢）。这些算法的复杂性来自于试图处理由于透视原理（相机附近的物体移动速度看起来比远端物体快）导致的不规则物体的运动；图像噪声；干扰物，例如场景中的其他脸；由于手或其他物体导致的脸部遮挡；光照变化。我们想要一种快速、高计算效率的算法，能够在操作中处理上述问题，即能在不减少计算效率的情况下减轻上述问题的影响。

In order, therefore, to find a fast, simple algorithm for basic tracking, we have focused on color-based tracking [[7][8][9][10][11]], yet even these simpler algorithms are too computationally complex (and therefore slower at any given CPU speed) due to their use of color correlation, blob and region growing, Kalman filter smoothing and prediction, and contour considerations. The complexity of the these algorithms derives from their attempts to deal with irregular object motion due to perspective (near objects to the camera seem to move faster than distal objects); image noise; distractors, such as other faces in the scene; facial occlusion by hands or other objects; and lighting variations. We want a fast, computationally efficient algorithm that handles these problems in the course of its operation, i.e., an algorithm that mitigates the above problems “for free.”

To develop such an algorithm, we drew on ideas from robust statistics and probability distributions. Robust statistics are those that tend to ignore outliers in the data (points far away from the region of interest). Thus, robust algorithms help compensate for noise and distractors in the vision data. We therefore chose to use a robust non-parametric technique for climbing density gradients to find the mode of probability distributions called the mean shift algorithm [2]. (The mean shift algorithm was never intended to be used as a tracking algorithm, but it is quite effective in this role.)

为了开发出这样一种算法，我们借鉴了稳健统计和概率分布方法。稳健统计趋于忽略数据中的异常值（远离感兴趣区域的点）。因此，稳健统计算法能够帮助弥补视觉数据中的噪音和干扰项造成的影响。因此，我们选择使用 mean shift 算法，一种具有鲁棒性的无参数方法，该方法通过求取密度梯度上升得到概率分布（mean shift 算法从来没有被用来作为一个跟踪算法，但它实际上在跟踪方面十分有效）。

The mean shift algorithm operates on probability distributions. To track colored objects in video frame sequences, the color image data has to be represented as a probability distribution [1]; we use color histograms to accomplish this. Color distributions derived from video image sequences change over time, so the mean shift algorithm has to be modified to adapt dynamically to the probability distribution it is tracking. The new algorithm that meets all these requirements is called CAMSHIFT.

mean shift 算法的操作是基于概率分布的。为了追踪视频帧序列中的彩色物体，彩色图像数据必须转化为概率分布的形式；我们使用了颜色直方图来完成这个目标。来自视频图像序列的颜色分布数据随时间变化，因此必须改进 mean shift 算法以动态适应它追踪的概率分布数据。这种能满足上述所有要求的算法被我们称为 CAMSHIFT.

For face tracking, CAMSHIFT tracks the X, Y, and Area of the flesh color probability distribution representing a face. Area is proportional to Z, the distance from the camera. Head roll is also tracked as a further degree of freedom. We then use the X, Y, Z, and Roll derived from CAMSHIFT face tracking as a perceptual user interface for controlling commercial computer games and for exploring 3D graphic virtual worlds.

CAMSHIFT 通过追踪代表脸的肤色概率分布数据的 X 坐标、Y 坐标和面积来实现脸部追踪。面积和 Z 坐标大小（距离摄像头的距离）成正比。头部的转动也会被作为另一个自由度进行跟踪。然后我们使用从 CAMSHIFT 脸部追踪得到的 X,Y,Z 轴和旋转轴数据来控制计算机游戏并探索虚拟的 3D 世界。

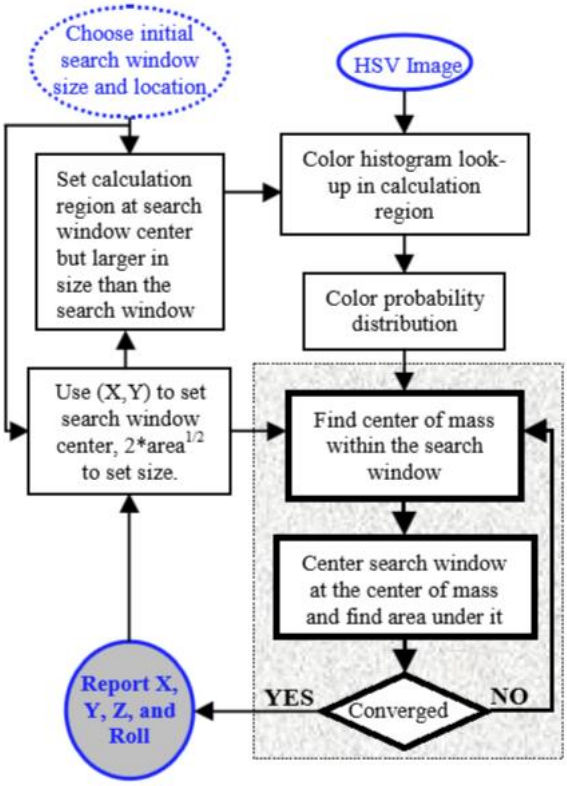


Figure 1: Block diagram of color object tracking

Figure 1: Block diagram of color object tracking

Figure 1 summarizes the algorithm described below. For each video frame, the raw image is converted to a color probability distribution image via a color histogram model of the color being tracked (flesh for face tracking). The center and size of the color object are found via the CAMSHIFT algorithm operating on the color probability image (the gray box is the mean shift algorithm). The current size and location of the tracked object are reported and used to set the size and location of the search window in the next video image. The process is then repeated for continuous tracking.

图一概括了下面描述的算法。对于视频中的每一帧，其原始图像被转化为通过被追踪物体（皮肤或者脸）的颜色直方图模型得到的颜色概率分布图。通过 CAMSHIFT 算法（灰色的框内为 mean shift 算法）对颜色概率图处理得到彩色物体的中心和大小。被追踪物体当前的大小和位置被记录并被用于设置下一帧视频图像的搜索框的大小和位置。然后重复该过程实现连续追踪。

Video Demonstrations

The following three videos demonstrate CAMSHIFT in action.

1. FaceTrack_Fast.avi
2. FaceTrack_Distractors.avi
3. FaceTrack_HandOcclusion.avi

The first video shows CAMSHIFT tracking rapid face movements. The second video shows CAMSHIFT tracking a face with other faces moving in the scene. The third video shows CAMSHIFT tracking a face through hand occlusions. These videos are available from this paper on the Web in the *Intel Technology Journal Q2'98* under the site http://developer.intel.com/technology/itj.

展示视频

下面三个视频将用来展示 CAMSHIFT。

1. FaceTrack_Fast.avi
2. FaceTrack_Distractors.avi
3. FaceTrack_HandOcclusion.avi

第一个视频展示了 CAMSHIFT 追踪快速移动的脸。第二个视频展示了 CAMSHIFT 在有其他脸在场景中运动时追踪目标脸。第三个视频展示了 CAMSHIFT 在有手遮挡的情况下对脸的追踪。这些视频可以在 http://developer.intel.com/technology/itj 上找到。

Color Probability Distributions

In order to use CAMSHIFT to track colored objects in a video scene, a probability distribution image of the desired color (flesh color in the case of face tracking) in the video scene must be created. In order to do this, we first create a model of the desired hue using a color histogram. We use the Hue Saturation Value (HSV) color system [5][6] that corresponds to projecting standard Red, Green, Blue (RGB) color space along its principle diagonal from white to black (see arrow in Figure 2). This results in the hexcone in Figure 3. Descending the V axis in Figure 3 gives us smaller hexcones corresponding to smaller (darker) RGB subcubes in Figure 2.

颜色概率分布

为了使用 CAMSHIFT 来跟踪视频场景中的彩色对象，必须创建视频场景中所需颜色（在脸部跟踪中为肤色）的概率分布图像。 为了做到这一点，我们首先使用颜色直方图创建所需色调的模型。 我们使用 HSV 颜色空间，其对应于沿其主对角线从白色到黑色投影标准的红色，绿色，蓝色（RGB）色彩空间（见图 2 中的箭头），形成了一个六角锥体。图 3 中的沿 V 轴

向下可以得到较小的六角锥体，对应于图 2 中较小（较深）的 RGB 子单元。
HSV space separates out hue (color) from saturation (how concentrated the color is) and from brightness. We create our color models by taking 1D histograms from the H (hue) channel in HSV space.
HSV 空间将色调（颜色）与饱和度（颜色集中度）和亮度分开。 我们通过从 HSV 空间中的 H（色调）通道获取一维直方图来创建我们的颜色模型。

For face tracking via a flesh color model, flesh areas from the user are sampled by prompting users to center their face in an onscreen box, or by using motion cues to find flesh areas from which to sample colors. The hues derived from flesh pixels in the image are sampled from the H channel and binned into an 1D histogram. When sampling is complete, the histogram is saved for future use. More robust histograms may be made by sampling flesh hues from multiple people.

为了实现通过肤色模型进行脸部跟踪，通过提示用户将他们的脸部居中放置在屏幕框中，或者使用运动检测找到肤色区域，再通过从中抽取颜色来对来自用户的肌肉区域进行采样。 从图像中的肤色像素点的 H 通道中得到采样值并装入一维直方图中。 当采样完成时，直方图保存以备将来使用。 可以通过采样多个人的肤色来得到更健壮的直方图。

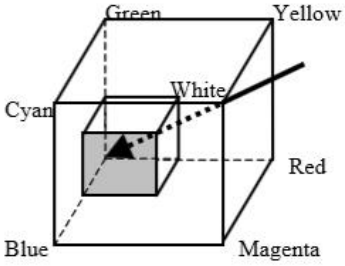


Figure 2: RGB color cube

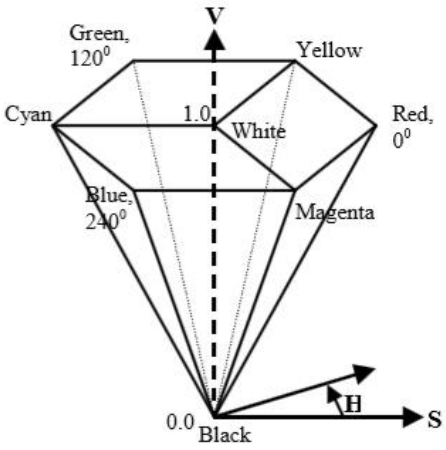


Figure 3: HSV color system

Even simple flesh histograms tend to work well with a wide variety of people without having to be updated. A common misconception is that different color models are needed for different races of people, for example, for blacks and whites. This is not true. Except for albinos, humans are all the same color (hue). Dark-skinned people simply have greater flesh color saturation than light-skinned people, and this is separated out in the HSV color system and ignored in our flesh-tracking color model.
即使简单的肤色直方图无需更新也能适用于很多人。一个常见的误解是，不同种族的人需要不同的颜色模型，例如黑人和白人。这并不是事实。除了白化病患者，所有人类肤色都是相同的颜色（色调）。黑皮肤的人的肤色只是比白皮肤的饱和度高，在 HSV 颜色系统中饱和度被分离出来，在我们的肉色跟踪颜色模型中被忽略。

During operation, the stored flesh color histogram is used as a model, or lookup table, to convert incoming video pixels to a corresponding probability of flesh image as can be seen in the right-hand image of Figure 6. This is done for each video frame. Using this method, probabilities range in discrete steps from zero (probability 0.0) to the maximum probability pixel value (probability 1.0). For 8-bit hues, this range is between 0 and 255. We then track using CAMSHIFT on this probability of flesh image.
在操作期间，存储的肤色直方图被用作模型或查找表，针对每个视频帧，将输入的视频像素转换成相应的肤色图像的概率，如图 6 所示。使用这种方法，得到从零（概率 0.0）到最大概率像素值（概率 1.0）的离散步骤中的概率范围。 对于 8 位的色调值，此范围在 0 到 255 之间。然后，我们根据肤色的概率使用 CAMSHIFT 进行跟踪。

When using real cameras with discrete pixel values, a problem can occur when using HSV space as can be seen in Figure 3. When brightness is low (V near 0), saturation is also low (S near 0). Hue then becomes quite noisy, since in such a small hexcone, the small number of discrete hue pixels cannot adequately represent slight changes in RGB. This then leads to wild swings in hue values. To overcome this problem, we simply ignore hue pixels that have very low corresponding brightness values. This means that for very dim scenes, the camera must auto-adjust or be adjusted for more brightness or else it simply cannot track. With sunlight, bright white colors can take on a flesh hue so we also use an upper threshold to ignore flesh hue pixels with corresponding high brightness. At very low saturation, hue is not defined so we also ignore hue pixels that have very low corresponding saturation (see Implementation Details section below).
当使用具有离散像素值的真实相机并使用如图 3 所示的 HSV 空间时，可能会出现问题。当亮度低（V 接近 0）时，饱和度也很低（S 接近 0）。 色调图像变得相当嘈杂，因为在这样小的六边形中，少数离散的色调像素不能充分表示 RGB 中的轻微变化。 这样会导致色调值的波动。 为了克服这个问题，我们忽略调对应亮度值非常低的色调像素。 这意味着对于非常暗淡的场景，相机必须自动或手动调整提高亮度，否则无法实现跟踪。 在阳光下，明亮的白色的色调和肤色接近，因此我们也可以使用上限阈值来忽略具有相应高亮度的肤色色调像素。 在非常低的饱和度下，色调不存在，所以我们也忽略了具有非常低的相应饱和度的色调像素（见下面的实施细节部分）。

Originally, we used a 2D color histogram built from normalized red green (r,g) space (r = R/(R+G+B), g = G/(R+G+B)). However, we found that such color models are much more sensitive to lighting changes since saturation (which is influenced by lighting) is not separated out of that model.
最初，我们使用由归一化的红绿（r，g）空间（r = R /（R + G + B），g = G /（R + G + B））构建的 2D 颜色直方图。 然而，我们发现，由于饱和度（受照明影响）不会从该模型中分离出来，所以这种颜色模型对照明变化非常敏感。

CAMSHIFT Derivation

The closest existing algorithm to CAMSHIFT is known as the mean shift algorithm [2][18]. The mean shift algorithm is a non-parametric technique that climbs the gradient of a probability distribution to find the nearest dominant mode (peak).
CAMSHIFT 的推导过程
现有的最接近 CAMSHIFT 算法被称为 mean shift 算法[2] [18]。 mean shift 算法是一种非参数化技术，可以通过概率分布的梯度上升法，找到最接近的主模式（peak）。

How to Calculate the Mean Shift Algorithm
1. Choose a search window size.
2. Choose the initial location of the search window
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).

如何实现 Mean Shift 算法
1. 选择搜索窗口的尺寸大小。
2. 选择搜索窗口的初始位置
3. 计算搜索窗口中的平均位置。
4. 将搜索窗口置于步骤 3 中计算的平均位置
5. 重复步骤 3 和 4 直到收敛（或直到平均位置移动小于预设阈值）

How to Calculate the Continuously Adaptive Mean Shift Algorithm
如何实现 CAMSHIFT 算法

1. Choose the initial location of the search window.
2. Mean Shift as above (one or many iterations); store the zeroth moment.
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).

1. 选择搜索框的初始位置。
2. 使用上面提到的 Mean Shift 算法得到零阶矩。
3. 设定搜索框使得其尺寸和第二步得到的零阶矩一致。
4. 重复二、三步，直到收敛（平均位置移动小于预设阈值）

In Figure 4 below, CAMSHIFT is shown beginning the search process at the top left step by step down the left then right columns until convergence at bottom right. In this figure, the red graph is a 1D cross-section of an actual sub-sampled flesh color probability distribution of an image of a face and a nearby hand. In this figure, yellow is the CAMSHIFT search window, and purple is the mean shift point. The ordinate is the distribution value, and the abscissa is the horizontal spatial position within the original image. The window is initialized at size three and converges to cover the tracked face but not the hand in six iterations. In this sub-sampled image, the maximum distribution pixel value is 206 so we set the width of the search window to be 2*M0/206 (see discussion of window size in the Implementation Details section below). In this process, CAMSHIFT exhibits typical behavior: it finds the center of the nearest connected distribution region (the face), but ignores nearby distractors (the hand).

在下面的图 4 中，显示了 CAMSHIFT 的过程，首先从左上角开始搜索过程，一步一步地向左下移动右边界，直到右下角收敛。在该图中，红图是面部和附近手的图像的实际的子采样肉色概率分布的一维横截面。在该图中，黄色是 CAMSHIFT 搜索窗口，紫色是平均移位点。纵坐标是分布值，横坐标是原始图像中的水平空间位置。窗口以三号大小初始化，并在六次迭代后收敛并覆盖跟踪的面部，而没有覆盖手。在这个子采样图像中，最大分布像素值为 206，所以我们将搜索窗口的宽度设置为 2 * M0 / 206（参见下面的实现细节部分的窗口大小的讨论）。在这个过程中，CAMSHIFT 表现出典型的特征行为：它找到最接近的分布区域（面）的中心，但忽略附近的干扰物（手）。

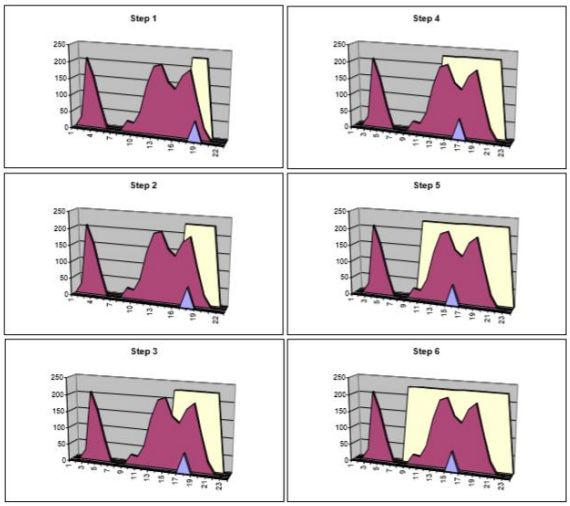


Figure 4: CAMSHIFT in operation down the left then right columns

Figure 4 shows CAMSHIFT at startup. Figure 5 below shows frame to frame tracking. In this figure, the red color probability distribution has shifted left and changed form. At the left in Figure 5, the search window starts at its previous location from the bottom right in Figure 4. In one iteration it converges to the new face center.
图 4 显示了启动时的 CAMSHIFT 算法。下面的图 5 显示了帧间跟踪。在该图中，红色概率分布向左移动并改变形式。 在图 5 的左侧，搜索窗口从图 4 中右下角的上一个位置开始。在一次迭代中，它收敛到新的面部中心。

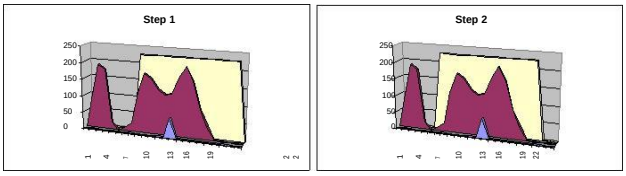


Figure 5: Example of CAMSHIFT tracking starting from the converged search location in Figure 4 bottom right

Mean Shift Alone Does Not Work
Mean Shift 算法无法单独工作

The mean shift algorithm alone would fail as a tracker. A window size that works at one distribution scale is not suitable for another scale as the color object moves towards and away from the camera. Small fixed-sized windows may get lost entirely for large object translation in the scene. Large fixed-sized windows may include distractors (other people or hands) and too much noise.

单独的平均偏移算法是无法作为跟踪器工作的。 在一个分布比例范围内工作的窗口尺寸不适用于随着彩色物体靠近和远离照相机移动情况。 小型固定大小的窗户可能会在场景中的大型物体遮挡中完全丢失。 大型固定尺寸的窗户可能包括干扰物（其他人或手）和导致过多的噪音。

CAMSHIFT for Video Sequences

When tracking a colored object, CAMSHIFT operates on a color probability distribution image derived from color histograms. 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.

针对视频序列的 CAMSHIFT 算法

当跟踪彩色对象时，CAMSHIFT 对从颜色直方图导出的颜色概率分布图像进行操作。 CAMSHIFT 计算其二维搜索窗口内二维颜色概率分布的质心，并以此重新设置窗口中心，然后计算下一窗口的大小。 因此，我们不需要计算整个图像的颜色概率分布，而是可以将分布的计算限制在当前 CAMSHIFT 窗口周围的较小图像区域。 当肤色不是主导图像时，这往往可以节省大量的计算资源。 我们将计算区域大小的反馈称为耦合 CAMSHIFT 算法。

How to Calculate the Coupled CAMSHIFT Algorithm
如何实现耦合 CAMSHIFT 算法

1. First, set the calculation region of the probability distribution to the whole image.
2. Choose the initial location of the 2D mean shift search window.
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.
4. Mean shift to convergence or for a set number of iterations. 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.

1. 首先，将概率分布的计算区域设置为整个图像。
2. 选择 2D mean shift 搜索窗口的初始位置。
3. 在稍大于 mean shift 搜索窗口大小的区域中，计算以搜索窗口位置为中心的 2D 区域的颜色概率分布。
4. 使用 mean shift 算法计算，直到收敛或设定迭代次数。 存储得到的零阶矩（区域或大小）和平均位置。
5. 对于下一个视频帧，将搜索窗口置于步骤 4 中存储的平均位置，并将窗口大小设置为得到的零阶矩，转到步骤 3 继续计算。

For each frame, the mean shift algorithm will tend to converge to the mode of the distribution. Therefore, CAMSHIFT for video will tend to track the center (mode) of color objects moving in a video scene. Figure 6 shows CAMSHIFT locked onto the mode of a flesh color probability distribution (mode center and area are marked on the original video image). In this figure, CAMSHIFT marks the face centroid with a cross and displays its search window with a box.

对于每一帧图像，mean shift 算法将倾向于收敛到分布的模式。 因此，用于视频的 CAMSHIFT 算法将倾向于跟踪在视频场景中移动的彩色对象的中心。 图 6 显示 CAMSHIFT 锁定在肤色颜色概率分布的模式（模式中心和区域在原始视频图像上标记）。 在该图中，CAMSHIFT 用十字标记脸部重心，并用一个框显示其搜索窗口。

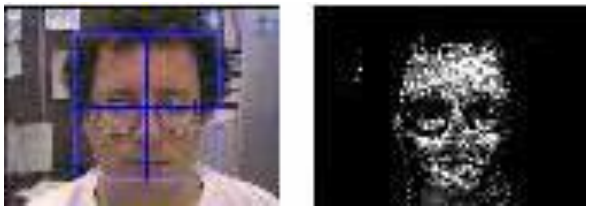


Figure 6: A video image and its flesh probability image

Calculation of Head Roll
计算头部旋转

The 2D orientation of the probability distribution is also easy to obtain by using the second moments during the course of CAMSHIFT's operation where (x,y) range over the search window, and I(x,y) is the pixel (probability) value at (x,y):

2D 概率分布图像的方向可以通过计算 CAMSHIFT 运行过程中搜索窗的二阶矩得到。下面式子中，(x,y)为坐标，I(x,y)为该坐标的像素值（即概率）。

Second moments are

$$M_{20} = \sum\sum x^2 I(x,y); \quad M_{20} = \sum\sum x^2 I(x,y).$$

Then the object orientation (major axis) is

$$\theta = \frac{\arctan\left(\left(\frac{M_{11}}{M_{00}} - x_c y_c\right)\Big/\left(\left(\frac{M_{20}}{M_{00}} - x_c^2\right) - \left(\frac{M_{02}}{M_{00}} - y_c^2\right)\right)\right)}{2}$$

The first two Eigenvalues (major length and width) of the probability distribution "blob" found by CAMSHIFT may be calculated in closed form as follows [4]. Let

通过 CAMSHIFT 发现概率分布封闭的"团块"的前两个特征值（主轴长度和宽度）可以通过下面的方式计算。令

$$a = \frac{M_{20}}{M_{00}} - x_c^2,$$

$$b = 2\left[\frac{M_{11}}{M_{00}} - x_c y_c\right]$$

and

$$c = \frac{M_{02}}{M_{00}} - y_c^2 ,$$

Then length l and width w from the distribution centroid are

然后计算质心的长度 l 和宽度 w

$$l = \sqrt{\frac{(a+c) + \sqrt{b^2 + (a-c)^2}}{2}} ,$$

$$w = \sqrt{\frac{(a+c) - \sqrt{b^2 + (a-c)^2}}{2}} .$$

When used in face tracking, the above equations give us head roll, length, and width as marked in Figure 7.

当在脸部追踪过程中，通过求解上述方程可以得到如图 7 中标记的头部旋转、长、宽。

Figure 7: Orientation of the flesh probability distribution marked on the source video image

CAMSHIFT thus gives us a computationally efficient, simple to implement algorithm that tracks four degrees of freedom (see Figure 8).

因此，CAMSHIFT 为我们提供了一种计算效率高，实现简单的跟踪四个自由度的算法。

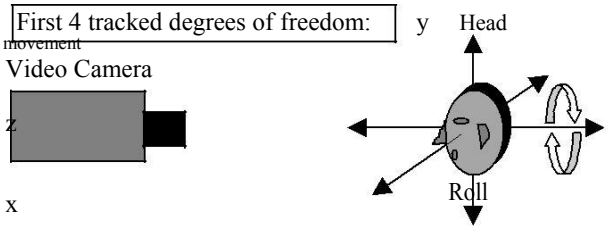


Figure 8: First four head tracked degrees of freedom: X, Y, Z location, and head roll

How CAMSHIFT Deals with Image Problems
CAMSHIFT 算法如何处理图像干扰问题

When tracking color objects, CAMSHIFT deals with the image problems mentioned previously of irregular object motion due to perspective, image noise, distractors, and facial occlusion as described below.

跟踪彩色对象时，CAMSHIFT 通过下述方式处理了由于透视效应，图像噪声，干扰物和面部遮挡引起的不规则物体运动所引起的图像问题。

CAMSHIFT continuously re-scales itself in a way that naturally fits the structure of the data. A colored object's potential velocity and acceleration scale with its distance to the camera, which in turn, scales the size of its color distribution in the image plane. Thus, when objects are close, they can move rapidly in the image plane, but their probability distribution also occupies a large area. In this situation, CAMSHIFT's window size is also large and so

CAMSHIFT 不断地以自然适合数据结构的方式重新调整自身。 彩色物体的相对速度和加速度随着摄像机的距离的改变而改变，反过来，它可以缩放其在图像平面中的颜色分布的大小。 因此，当物体靠近时，它们可以在图像平面中快速移动，但它们的概率分布也占据了大面积。 在这种情况下，CAMSHIFT 的窗口大小也很大，因此在目标位移较大时仍可以捕获目标。

can catch large movements. When objects are distant, the color distribution is small so CAMSHIFT's window size is small, but distal objects are slower to traverse the video scene. This natural adaptation to distribution scale and translation allows us to do without predictive filters or variables–a further computational saving–and serves as an in-built antidote to the problem of erratic object motion.
当物体距离较远时，颜色分布面积很小，因此 CAMSHIFT 的搜索窗口很小，但是远端的物体要穿越视频场景较慢。 这种对缩放和变化的自然适应允许我们在没有预测过滤器或变量的情况下节省更多的计算资源，并且能够实现对不规则运动的追踪。

CAMSHIFT's windowed distribution gradient climbing causes it to ignore distribution outliers. Therefore, CAMSHIFT produces very little jitter in noise and, as a result, tracking variables do not have to be smoothed or filtered. This gives us robust noise tolerance.
CAMSHIFT 对窗口内的概率分布进行梯度上升法求取局部最大值导致它忽略分布异常值。 因此，CAMSHIFT 在噪声中产生很小的抖动，因此跟踪变量不必平滑或过滤。 这使得算法有较好的鲁棒性和噪声容忍性。

CAMSHIFT's robust ability to ignore outliers also allows it to be robust against distractors. Once CAMSHIFT is locked onto the mode of a color distribution, it will tend to ignore other nearby but non-connected color distributions. Thus, when CAMSHIFT is tracking a face, the presence of other faces or hand movements in the scene will not cause CAMSHIFT to loose the original face unless the other faces or hand movements substantially occlude the original face.
CAMSHIFT 强大的忽视异常值的能力也使得它能够有效地对抗干扰。 一旦 CAMSHIFT 锁定到颜色分布的模式，它将倾向于忽略其他附近的但不连接的颜色分布。因此，当 CAMSHIFT 正在跟踪脸部时，场景中其他脸部或手部动作的存在不会导致 CAMSHIFT 丢失目标脸部，除非其他脸部或手部动作遮挡了目标脸部。

Moreover, when CAMSHIFT's window size is set somewhat greater than the root of the distribution area under its window, CAMSHIFT tends to grow to encompass the connected area of the distribution that is being tracked (see Figure 4). This is just what is desired for tracking whole objects such as faces, hands, and colored tools. This property enables CAMSHIFT to not get stuck tracking, for example, the nose of a face, but instead to track the whole face.
此外，当 CAMSHIFT 的窗口大小设置为稍大于其窗口下的分布区域的根部时，CAMSHIFT 趋向于增长以包含被跟踪的分布的连接区域（参见图 4）。 这是追踪整个目标，如脸，手和彩色工具所需要的。 该属性使 CAMSHIFT 不会只跟踪目标的一部分，例如，只追踪脸部的鼻子，而不是跟踪整个脸部。

CAMSHIFT'S Use as a Perceptual Interface

Treatment of CAMSHIFT Tracking Variables for Use in a Perceptual User Interface
用于感知用户界面的 CAMSHIFT 跟踪变量的处理

Figure 8 above shows the variables X, Y, Z, and Roll returned by the CAMSHIFT face tracker. 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.
上面的图 8 显示了 CAMSHIFT 面部追踪器返回的变量 X，Y，Z 和 Roll。 对于游戏和图形控制，X，Y，Z 和 Roll 通常需要"中点"位置; 也就是相对于测量进一步面部移动的位置。 例如，如果拍摄的视频图像具有 120×160 的尺寸（Y，X），则典型的中点位置可以是 Y = 60，X = 80。 那么如果 X <80，用户已经移动了 80-X; 如果 X> 80，用户已经移动了 X-80 等等。

Piecewise Linear Transformation of Control Variables
控制变量的分段线性变换

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.
为了在包括抖动阻尼中间运动区域的各个位置处获得差分控制，每个变量的相对运动（高于或低于中间位置）被缩放为"N"个不同的范围。

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.
在上面的 X 变量示例中，如果 X 在范围＃1 中，X 将按"X x 1"缩放; 如果 X 在范围＃2 中，X 将按"X x 2"缩放，依此类推。

The formula for mapping captured video head position P to a screen movement factor F is
将捕获的视频头位置 P 映射到屏幕移动因子 F 的公式为

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

where $[\#]^+$ equals "#" if # > 0, and zero otherwise; min(A, B) returns the minimum of A or B; b_1-b_N represents the bounds of the ranges, s_1-s_N are the corresponding scale factors for each range, and P is the absolute value of the difference of the variable's location from neutral.
符号[#]+在#》0 时等于#，否则等于 0；min（A，B）返回 A 或 B 的最小值；b1-bN 表示边界范围，s1-sN 是每个范围的相应比例因子，P 是变量的位置与中值的差的绝对值。

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 Figure 9.

如图 9 所示，以分段线性方式，随着离开中性线的距离增加时运动加速度的增加，可以增强中间位置的稳定性。

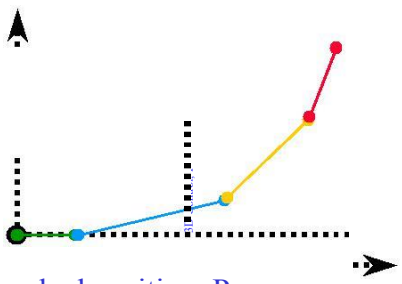


Figure 9: Piecewise linear transformation of CAMSHIFT position P interface control variable F

Frame Rate Adjustment
帧率调整

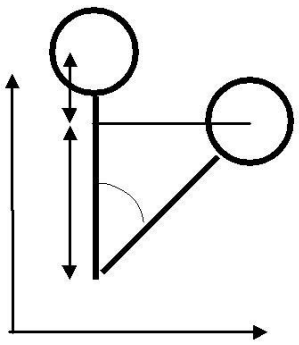
If the graphic's rendering rate R can be determined, the final screen movement S is
如果可以确定图像渲染速率 R，最终的屏幕移动量 S 为

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. In computer-rendered graphic or game scenes, simple views (for example, looking up at blue sky) are rendered much faster than complex views (for example, texture mapped city skylines). The final rate of movement should not depend on the complexity of the 3D view if one wants to achieve satisfying motion when immersed in a 3D scene.
计算机图形和游戏移动命令 S 可以在每个渲染图形的帧上更新。 因此，最好使移动量 S 对帧率敏感。 在计算机渲染的图形或游戏场景中，简单的视图（例如，仰望蓝天）比复杂视图（例如，纹理映射的城市天际线）渲染得快得多。 如果想要在沉浸式 3D 场景中实现令人满意的运动，最终的运动速度不应该取决于 3D 视图的复杂性。

Y Variable Special Case for Seated User
针对坐着的用户的特殊 Y 轴变量

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 10 below.
如果用户坐着面对相机（坐标在 X、Y 轴的中部），然后通过在他/她的椅子上转动左右倾斜（X 轴上的位移），Y 轴数据会如下方图 10 所示减小。



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. Empirically, the ratio of the 2nd Eigenvector to torso length from face centroid is
为了克服这一点，我们使用经验观察，局部面部肤色分布长度的第二特征值（面半宽度）与面部尺寸成比例，平均来说，这通常与身体尺寸成比例 。 经验上，第二特征向量与身体重心的躯干长度之比为 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 10 so that sin(A) = x/13.75. Then,
给定短距离 x（以第二特征向量为单位），并且坐标尺寸为 13.75，如图 10 所示，使得 sin（A）= x / 13.75。 然后可得，

A = sin-1(x/13.75), (control equation 4)

so c = 13.75cos(A), and

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

in units of 2nd Eigenvectors. This is the Y distance to correct for (add back) when leaning.
该值以第二特征向量为单位。 这是在倾斜时要纠正（加回）的 Y 距离。

Roll Considerations
旋转轴计算注意事项

As can be seen from Figure 10, 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:
从图 10 可以看出，对于坐着的使用者而言，倾斜也引起头部旋转轴方向上的变化。因此，对于依赖于头部旋转轴的控制，应该对该倾斜的旋转轴进行校正。 校正可以通过两种方式完成：

1. 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.

2. Subtract the measured roll from the lean-induced roll, A, calculated in control Equation 4 above.

1. .使控制方程1中的第一个范围边界B1大到足以包含由于倾斜产生的面部朝向的变化。 然后使用比例值 s1 = 0，以使倾斜不产生旋转轴的变化。
2. 从上述对照等式4中计算的，从倾斜的旋转量中减去测量的旋转量。

Another possible problem can result when the user looks down too much as shown in Figure 11. In this case, the user is looking down at the keyboard. 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.

如图 11 所示，当用户向下方看时，可能会出现另一个问题。在这种情况下，用户正在看着键盘。 头向下低太多会导致前额成为主导追踪对象，反过来会导致脸部肤色团块看起来像水平方向。

Figure 11: Extreme down head pitch causes a corrupted head roll value

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:

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

为了纠正这些问题，我们定义了一个新的变量 Q，称为“旋转质量”.Q 是 CAMSHIFT 搜索窗口中分布颜色团块的前两个特征值，长度 l 和宽度 w 之间的比值：

For problem views of the face such as in Figure 11, we observe that Roll Quality is nearly 1.0. So, for face

针对图 11 中的出现的问题，我们观察到此时旋转质量接近 1.0。

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.
所以当进行脸部追踪时，旋转质量小于 1.25 的应被忽略（视为垂直方向的移动）。大于 2.0 的旋转质量也应该忽略掉，因为这种分布是不具备脸部特征的，而有可能是由于噪声或阻挡引起的。

CAMSHIFT's Actual Use as an Interface

CAMSHIFT 作为接口的实际应用

CAMSHIFT is being used as a face tracker to control games and 3D graphics. By inserting face control variables into the mouse queue, we can control unmodified commercial games such as Quake 2 shown in Figure 12. We used left and right head movements to slide a user left and right in the game, back and forth head movements to move the user backwards and forwards, up or down movements to let the user shoot (as if ducking or getting jolted by the gun), and 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.
CAMSHIFT 被用作面部追踪器来控制游戏和 3D 图形世界。 通过将面部控制变量插入到鼠标队列中，我们可以控制未经修改的商业游戏，如图 12 所示的 Quake 2。我们使用头部左右移动的动作在游戏中左右移动游戏角色，头部前后移动来使游戏角色前后移动，向上或向下移动以让角色开枪（就像躲避或者被枪的后坐力影响），并用向左或向右滚动的动作以在游戏中向左或向滑动游戏角色视角。 该方法已广泛应用于 30 多个不同用户的一系列演示中。

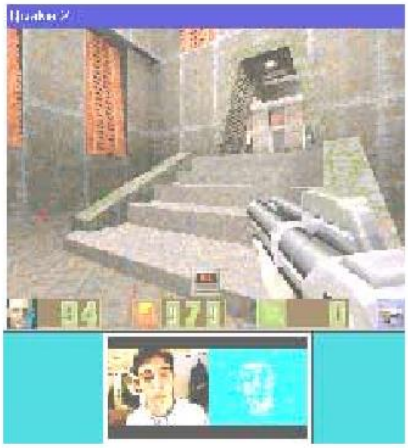


Figure 12: CAMSHIFT-based face tracker used to play Quake 2 hands free by inserting control variables into the mouse queue

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 13. Most users find it an enjoyable experience in which they naturally pick up how to control the graphics viewpoint movement.
通过 CAMSHIFT 实现的头部跟踪也被用于实验沉浸式 3D 控制，转换其中头部的自然移动以移动相应的 3D 图形相机视角。我们已经使用中国紫禁城和如图 13 所示的夏威夷大岛的 3D 图形模型进行了大量的测试。大多数用户认为这是一个愉快的经历，他们自然地掌握了控制图形相机视角运动的方法。

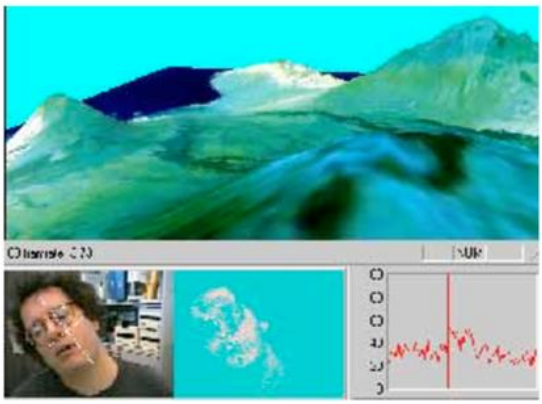


Figure 13: CAMSHIFT-based face tracker used to "fly" over a 3D graphic's model of Hawaii

CAMSHIFT Analysis
CAMSHIFT 性能分析

Comparison to Polhemus
和 Polhemus 公司的追踪器比较

In order to assess the tracking accuracy of CAMSHIFT, we compared its accuracy against a Polhemus tracker. Polhemus is a magnetic sensor connected to a system that measures six degrees of spatial freedom and thus can be used for object tracking when tethered to an object. 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 (see Figure 14a-d). The coordinate systems of Polhemus and the camera were carefully aligned prior to testing. The object tracked was pulled on a cart in a set trajectory away from the Polhemus origin. The comparison

为了评估 CAMSHIFT 的跟踪精度，我们将其准确度与 Polhemus 公司的跟踪器进行了比较。Polhemus 追踪器是一种连接到测量六度空间自由度系统的磁传感器，因此可以对绑定该传感器的对象进行跟踪。Polhemus 在空间位置内的观测精度为+/- 1.5cm，在 Polhemus 天线的 30 英寸内旋转观测精度约为 2.5°。

between CAMSHIFT and Polhemus in each of X, Y, Z, and Roll yielded the results shown Table 1. 我们将 Polhemus 跟踪器与使用 320x240 图像大小的 CAMSHIFT 颜色对象跟踪进行了比较（见图 14a-d）。我们在测试前仔细对齐了 Polhemus 追踪器和摄像机的坐标系，被追踪的对象被设置为到一个远离 Polhemus 追踪器原点的一组轨道上的推车上。两者在 X、Y、Z、ROLL 轴上的追踪结果差异如表一所示。

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

Table 1: Standard deviation of Polhemus vs. CAMSHIFT tracking differences

Z exhibited the worst difference because CAMSHIFT determines Z by measuring color area, which is inherently noisy. X, Y, and Roll are well within Polhemus's observed tracking error and therefore indistinguishable. Z is about 2cm off. Except for Z, these results are as good or better than much more elaborate vision tracking systems [17], although CAMSHIFT does not yet track pitch and yaw.

Z 坐标表现出差异最明显，因为 CAMSHIFT 通过测量颜色区域来确定 Z 坐标，这本质上是嘈杂的。 X，Y 和 Roll 坐标在 Polhemus 观察到的跟踪误差范围内，因此差异较小。 Z 坐标差距大概在 2 厘米左右。 除了 Z 轴，这些结果与更复杂的视觉跟踪系统一样好或更好[17]，尽管 CAMSHIFT 尚未跟踪俯仰和偏航。

噪声干扰下的追踪

Camshift 算法的鲁棒性和使用跟踪动态变化的概率分布模式也使它在有噪声干扰时仍有很好的跟踪性能。我们录制了一个头部运动视频序列，然后添加了 0,10,30 和 50％的均匀噪声。图 15 示出添加到原始图像的左边 50％的噪声，和在右边产生的颜色概率分布。 请注意，使用颜色模型可大大减少随机噪声，因为颜色噪声具有肤色特征的概率较低。然而，由于肤色模型被高度简化，在颜色概率分布图像中存在许多虚假的肤色像素。但是如图 16a-d 所示，CAMSHIFT 仍然能够在有高达 30％的白噪声干扰下跟踪 X，Y 和 Roll 轴数据。 Z 轴数据的问题更为明显，因为 CAMSHIFT 通过在其搜索窗口下跟踪分布区域来测量 Z 坐标，并且在图 15 中可以看到该区域受到噪声的高度影响。由于较窄的下巴区域噪音比较宽的前额更多，Y 轴测量值上出现了向上的漂移。 转动轴的追踪效果很好，直到噪音使脸部颜色分布的长度和宽度被遮蔽时才出现追踪问题。 因此，CAMSHIFT 无需额外的滤波或自适应平滑来处理噪声。

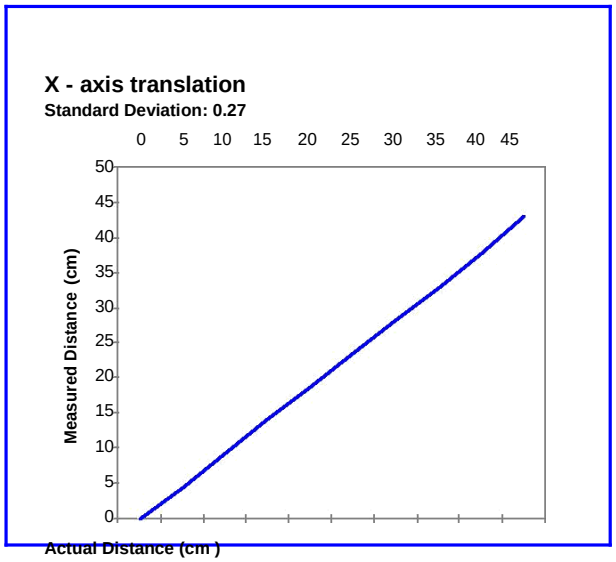


Figure 14a: Comparision of X tracking accuracy

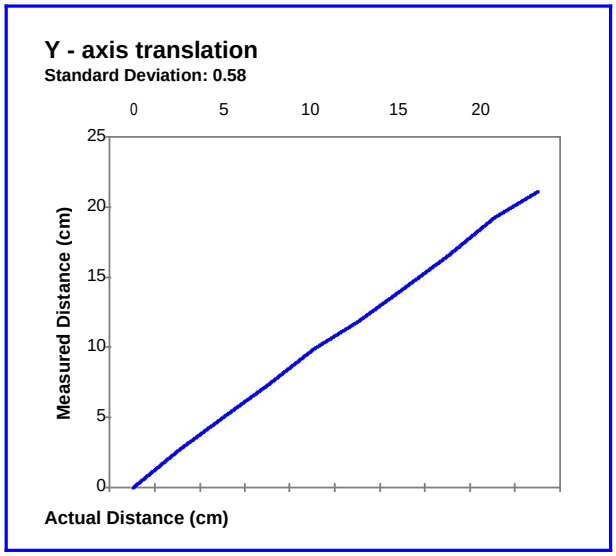


Figure 14b: Comparision of Y tracking accuracy

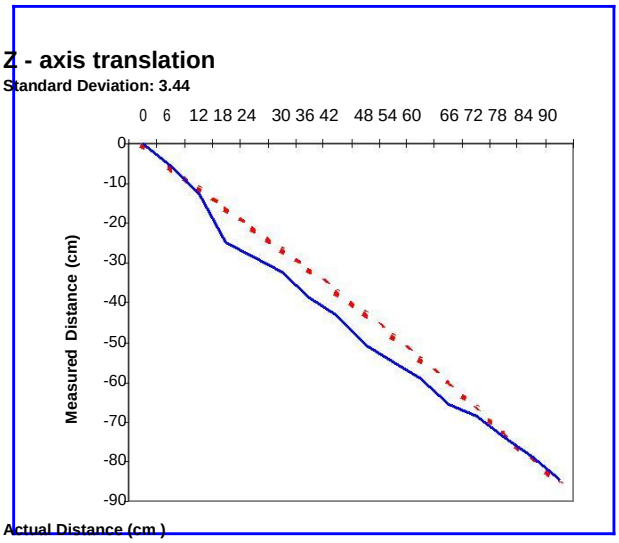


Figure 14c: Comparision of Z tracking accuracy

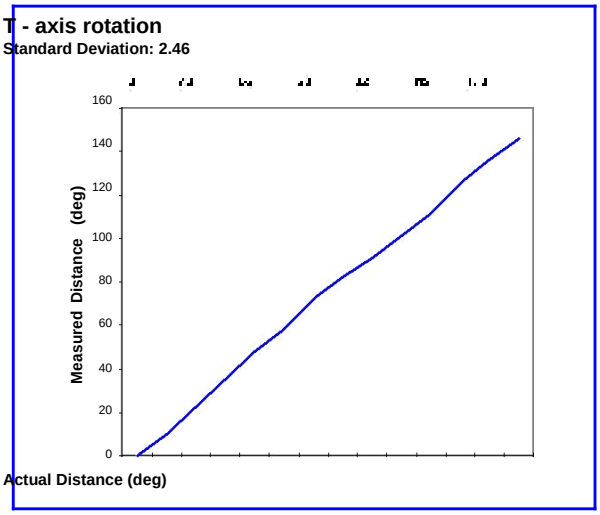


Figure 14d: Accuracy comparison of Polhemus and CAMSHIFT tracking for roll.

Tracking in Noise

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. Figure 15 shows 50% noise added to the raw image on the left, and the resulting color probability distribution on the right. Note that the use of a color model greatly cuts down the random noise since color noise has a low probability of being flesh color.

Nevertheless, the flesh color model is highly degraded and there are many spurious flesh pixels in the color probability distribution image. But CAMSHIFT is still able to track X, Y, and Roll quite well in up to 30% white noise as shown in Figure 16a-d. Z is more of a problem because CAMSHIFT measures Z by tracking distribution area under its search window, and one can see in Figure 15 that area is highly effected by noise. Y shows an upward shift simply because the narrower chin region exhibits more degradation in noise than the wider forehead. Roll tracks well until noise is such that the length and width of the face color distribution are obscured. Thus, CAMSHIFT handles noise well without the need for extra filtering or adaptive smoothing.

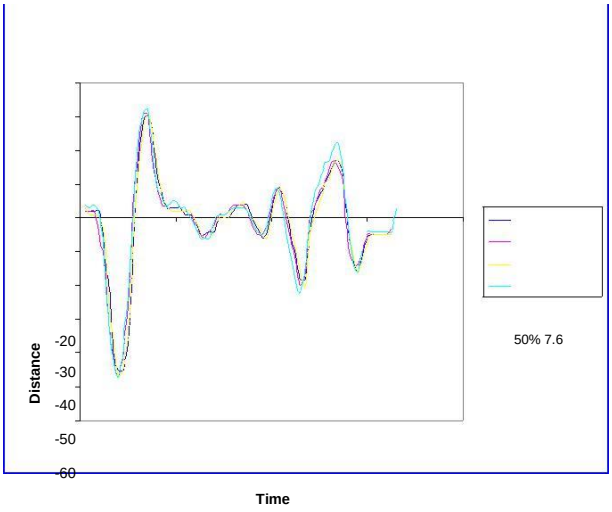


Figure 16a: X accuracy in 0-50% uniform noise

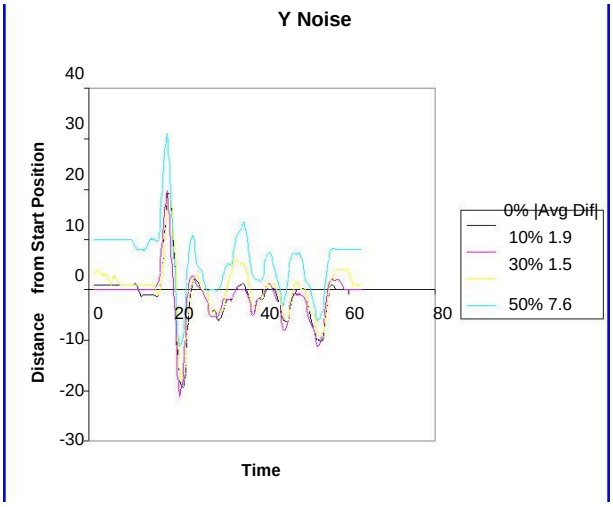


Figure 16b: Y accuracy in 0 - 50% uniform noise

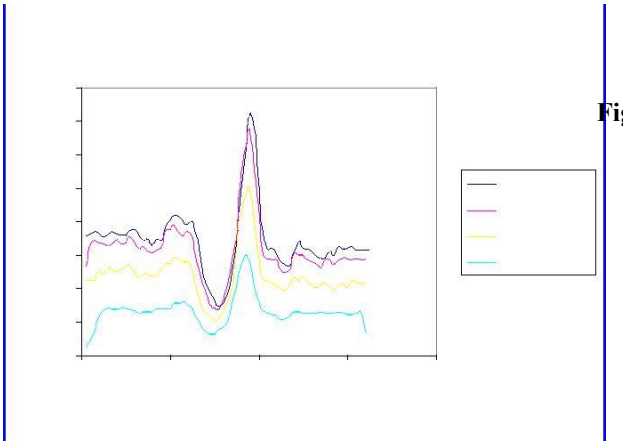


Figure 15: Tracl

Figure 16c: Z accuracy in 0-50% uniform noise

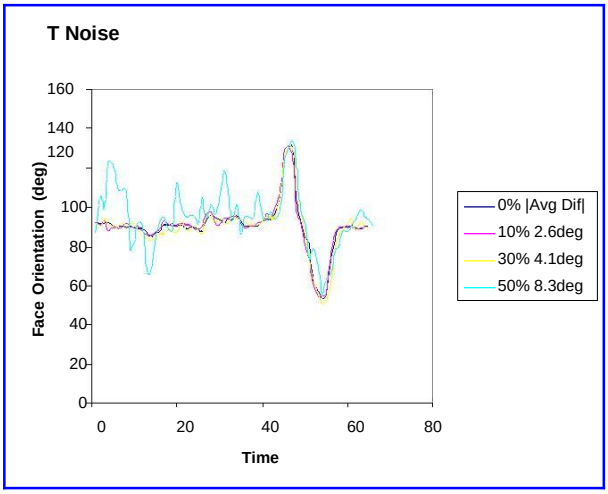


Figure 16d: Roll accuracy in 0-50% uniform noise

Tracking in the Presence of Distractions

CAMSHIFT's search window converges to span the nearest dominant connected probability distribution. If we adjust the nature of the probability distribution by properly setting the HSV brightness and saturation threshold (see Implementation Details section above), the search window will tend to stay just within the object being tracked as shown in the marked image at top left in Figure 17. In such cases, CAMSHIFT is robust against distracting (nearby) distributions and transient occlusions. This robustness occurs for distractors because the search window rarely contains the distractor as shown sequentially down the left, then right, columns of Figure 17.

CAMSHIFT 的搜索窗口收敛到最近的连通概率分布团块。 如果我们通过适当地设置 HSV 中的亮度和饱和阈值来调整概率分布的性质，则搜索窗口将倾向于保持在被跟踪的对象内，如图中左上角的标记图像所示，在这种情况下，干扰物对 CAMSHIFT 造成的影响有限。这种鲁棒性体现在干扰因素上，因为搜索窗口很少包含如图 17 左侧，右侧列所示的干扰物。

Figure 17: Tracking a face with background distractor faces (sequence: down left then right columns)

Table 2 shows the results collected from 44 sample point on five tracking runs with active background face distraction such as that shown in Figure 17. Since the distracting face rarely intersects much of CAMSHIFT's search window, the X, Y, and Z tracking variables are perturbed very little. 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.

表 2 显示了从五个跟踪运行过程中的 44 个采样点收集的结果。由于干扰物很少与 CAMSHIFT 的搜索窗口相交，所以 X，Y 和 Z 轴受到的影响较小。相比而言，旋转轴受到的影响更大，因为即使 CAMSHIFT 的搜索窗口中的干扰物的小交点也可以改变通过 CAMSHIFT 测量得到的肤色像素的有效方向。

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: Perturbation of CAMSHIFT tracking variables by face distractors

Figure 18: Tracking a face in the presence of passing hand occlusions (sequence: down left then right columns)

Table 3 shows the results collected from 43 sample points on five tracking runs with active transient hand occlusion of the face. Average perturbation is less than three pixels for X, Y, and Z. Roll is more strongly effected due to the arbitrary orientation of the hand as it passes through the search window.

表 3 显示了从五个跟踪运行过程中的 43 个采样点收集的结果，在测试过程中主动用手临时遮挡面部。对于 X，Y 和 Z，平均扰动小于三个像素。由于手通过搜索窗口时方向随机，因此更强烈地影响滚动轴数据。

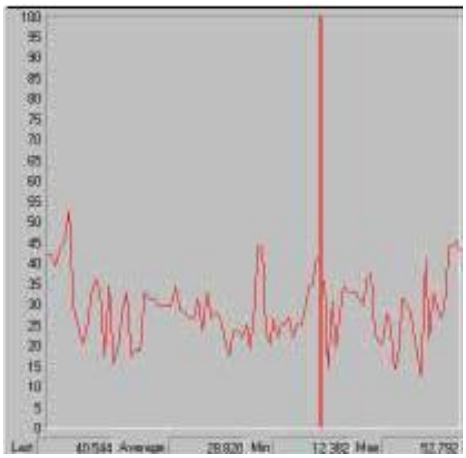


Figure 20: In actual use, CAMSHIFT consumed an average of 29% of one 300 MHz Pentium® II CPU when used to control a 3D graphic's Hawaii fly through

Discussion

This paper discussed a core tracking module that is part of a larger effort to allow computers to track and understand human motion, pose, and tool use. As such, the module was designed to be simple and computationally efficient. Yet, this core module must still handle the basic computer-vision problems outlined in this paper. We've seen that CAMSHIFT handles these problems as follows:
本文讨论了一个核心跟踪模块，这对计算机追踪、理解人的运动、姿势和工具的使用有很大影响。该模块设计简单，计算效率高。然而，这个核心模块还必须处理本文中概述的基本的计算机视觉问题。我们已经看到 CAMSHIFT 处理了下列问题：

1. **Irregular object motion**: CAMSHIFT scales its search window to object size thus naturally handling perspective-induced motion irregularities.
2. **Image noise**: The color model eliminates much of the noise, and CAMSHIFT tends to ignore the remaining outliers.
3. **Distractors:** CAMSHIFT ignores objects outside its search window so objects such as nearby faces and hands do not affect CAMSHIFT's tracking.
4. **Occlusion:** As long as occlusion isn't 100%, CAMSHIFT will still tend to follow what is left of the objects' probability distribution.
5. **Lighting variation:** Using only hue from the HSV color space and ignoring pixels with high or low brightness gives CAMSHIFT wide lighting tolerance.

1. 物体的不规则运动：CAMSHIFT 将其搜索窗口缩放到对象的大小，从而自然地处理由于透视原理引起的运动不规则问题。
2. 图像噪声：基于颜色的模型消除了大量的噪声，CAMSHIFT 倾向于忽略剩余的异常值。
3. 干扰物：Camshift 忽略搜索窗以外的物体，搜索窗附近的脸和手不影响 CamShift 跟踪。
4. 阻挡：只要没有发生 100％的阻挡，CAMSHIFT 仍然会倾向于跟随对象概率分布的剩余部分。
5. 照明变化：仅使用 HSV 色彩空间的色调，忽略高或低亮度的像素，CAMSHIFT 对宽范围的光照条件都有较好的容忍性。

CAMSHIFT's simplicity does cause limitations however. Since CAMSHIFT derives Z from object area estimates, Z is subject to noise and spurious values. The effects of noise are evident in Figure 16c. That CAMSHIFT can get spurious area values is evident in Figure 11.
CAMSHIFT 的简单性确实对追踪性能造成了限制。 由于 CAMSHIFT 从估计的对象区域中导出 Z 轴坐标，所以 Z 轴数据受到噪声和杂散值的影响较大。在图 16c 中可以看到明显的噪声影响，这导致 CAMSHIFT 会计算出到错误的面积值，如图 11 所示。

Since CAMSHIFT relies on color distributions alone, errors in color (colored lighting, dim illumination, too much illumination) will cause errors in tracking. More sophisticated trackers use multiple modes such as feature tracking and motion analysis to compensate for this, but more complexity would undermine the original design criterion for CAMSHIFT.
由于 CAMSHIFT 仅依赖颜色分布，错误的颜色数据（彩色照明，昏暗照明，过度曝光）将导致跟踪错误。 更复杂的跟踪器使用多种模式，如特征跟踪和运动分析来弥补这一点。但是更多的复杂性将破坏 CAMSHIFT 的原始设计标准。

Conclusion
结论

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], and it tracks well in noisy environments. Thus, as we have shown, even though CAMSHIFT was conceived as a simple part of a larger tracking system, it has many uses right now in game and 3D graphics' control.
CAMSHIFT 是一个简单、计算效率高的面部和彩色对象跟踪器。虽然由于其简单性带来了一些限制，我们仍然可以看到，CAMSHIFT 实际跟踪效果和更昂贵的 Polhemus 跟踪器（Polhemus）或更复杂的，计算量更大的视觉系统[17]一样，并且在嘈杂的环境中保持良好的跟踪能力。因此，正如我们所看到的，即使 CAMSHIFT 被认为是更大的跟踪系统的一个简单部分，但现在它在游戏和 3D 图形界面的控制中有很大的应用价值。

Adding perceptual interfaces can make computers more natural to use, more fun for games and graphics, and a better medium of communication. These new features consume more MIPs and so will take advantage of more MIPs available with future Intel® CPUs.
添加感知接口可以使计算机的使用更加自然，使游戏和图形界面更有趣，是一种更好的通信媒介。 这些新功

会消耗更多的运算性能，因此将来可以使用未来英特尔®的有更高 MIPS 的 CPU。

In this project, we designed a highly efficient face tracking algorithm rather than a more complex, higher MIPs usage algorithm. This was done because we want to be able to demonstrate compelling applications and interfaces on today's systems in order to prepare the way for the future use of computer vision on PCs. CAMSHIFT is usable as a visual interface now, yet designed to be part of a more robust, larger tracking system in the future. CAMSHIFT will be incorporated into larger, more complex, higher MIPs-demanding modules that provide more robust tracking, posture understanding, gesture and face recognition, and object understanding. 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.

在这个项目中，我们设计了一个高效的面部跟踪算法，而不是一个更复杂，占用更多 CPU 性能的算法。 这样做是因为我们希望能够在目前的系统上展示令人信服的应用程序和接口，以便为将来在 PC 上应用计算机视觉做好准备。 CAMSHIFT 现在可用于可视界面，而且设计为将来成为更健壮，更大型跟踪系统的一部分。 CAMSHIFT 将纳入更大，更复杂，更高性能要求的模块中，提供更强大的跟踪，姿势理解，姿态和面部识别以及目标解析。 这样，随着 Intel CPU 速度的增加，计算机视觉界面的功能将会增加。 因此，用户将能够通过升级到更高速度的 Intel CPU 来升级他们的计算机视觉界面。

Acknowledgments

Many thanks to Mark Holler for countless discussions and feedback leading to the development of this work. Special thanks to Ryan Boller for rapid implementation of the CAMSHIFT game control demo and for major help in implementing the testing routines for CAMSHIFT.

非常感谢 Mark Holler 无数次的讨论和反馈，使这项工作得以进行。特别感谢 Ryan Boller 提供的可快速部署的 CAMSHAFT 游戏控制演示，并为部署 CAMSHIFT 的测试例程提供了重要帮助。

References

1. D. Comaniciu and P. Meer, "Robust Analysis of Feature Spaces: Color Image Segmentation," CVPR'97, pp. 750-755.
2. K. Fukunaga, "Introduction to Statistical Pattern Recognition," Academic Press, Boston, 1990.
3. MMX™ technology optimized libraries in image, signal processing, pattern recognition and matrix math can be downloaded from http://developer.intel.com/design/perftool/perflibst/index.htm)
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.
5. A.R. Smith, "Color Gamut Transform Pairs," SIGGRAPH 78, pp. 12-19, 1978.
6. J.D. Foley, A. van Dam, S. K. Feiner and J.F. Hughes, "Computer graphics principles and practice," Addison-Wesley, pp. 590-591.
7. P. Fieguth and D. Terzopoulos, "Color-based tracking of heads and other mobile objects at video frame rates," In Proc. Of IEEE CVPR, pp. 21-27, 1997.
8. C. Wren, A. Azarbayejani, T. Darrell, A.Pentland, "Pfinder: Real-Time Tracking of the Human Body," SPIE Vol. 2615, 1995.
9. M. Hunke and A. Waibel, "Face locating and tracking for human-computer interaction," Proc. Of the 28 th Asilomar Conf. On Signals, Sys. and Comp., pp. 1277-1281, 1994.
10. K. Sobottka and I. Pitas, "Segmentation and tracking of faces in color images," Proc. Of the Second Intl. Conf. On Auto. Face and Gesture Recognition, pp. 236-241, 1996.
11. M. Swain and D. Ballard, "Color indexing," Intl. J. of Computer Vision, 7(1) pp. 11-32, 1991.
12. M. Kass, A. Witkin D.Terzopoulos, "Snakes: Active contour Models," Int. J. o f Computer Vision (1) #4, pp. 321-331, 1988.
13. C. Vieren, F. Cabestaing, J. Postaire, "Catching moving objects with snakes for motion tracking," Pattern Recognition Letters (16) #7, pp. 679-685, 1995.
14. A. Pentland, B. Moghaddam, T. Starner, "View-based and Modular Eigenspaces for face recognition," CVPR'94, pp. 84-91, 1994.
15. M. Isard, A. Blake, "Contour tracking by stochastic propagation of conditional density," Proc. 4 th European Conf. On Computer Vision, Cambridge, UK, April 1996.
16. T. Maurer, and C. von der Malsburg, "Tracking and learning graphs and pose on image sequence of faces," Proc. Of the Second Intl. Conf. On Auto. Face and Gesture Recognition, pp. 176-181, 1996.
1. 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.
2. Y. Cheng, "Mean shift, mode seeking, and clustering," IEEE Trans. Pattern Anal. Machine Intell., 17:790-799, 1995.

Author's Biography
作者简介

Dr. Gary R. Bradski received a Ph.D. in pattern recognition and computer vision from Boston University. He works in computer vision research in the Microcomputer Research Labs at Intel's Mission College campus. His interests include segmenting and tracking people in visual scenes; perceptual computer interfaces; applying visual 3D structure to 3D graphics; pattern recognition; biological perception and self-organization. His e-mail is gary.bradski@intel.com.

Gary R. Bradski 获波士顿大学模式识别和计算机视觉博士学位。他在英特尔社区大学的微机研究实验室做计算机视觉研究。他的专长包括在视觉场景中分割和跟踪人物；感知计算机接口；将视觉 3D 结构应用于 3D 图形；模式识别；生物感知和自组织。他的电子邮件是 gary.bradski@intel.com。