Foreground Segmentation Using Adaptive Mixture Models in Color and Depth

Michael Harville
Hewlett-Packard Labs
1501 Page Mill Rd., Palo Alto, CA 94304
harville@hpl.hp.com

Gaile Gordon, John Woodfill
Tyzx Inc.
301 Bryant Street, Palo Alto, CA 94301
gaile,john@tyzx.com

Abstract

Segmentation of novel or dynamic objects in a scene, often referred to as "background subtraction" or "foreground segmentation", is a critical early in step in most computer vision applications in domains such as surveillance and human-computer interaction. All previously described, real-time methods fail to handle properly one or more common phenomena, such as global illumination changes, shadows, inter-reflections, similarity of foreground color to background, and non-static backgrounds (e.g. active video displays or trees waving in the wind). The recent advent of hardware and software for real-time computation of depth imagery makes better approaches possible. We propose a method for modeling the background that uses per-pixel, time-adaptive, Gaussian mixtures in the combined input space of depth and luminance-invariant color. This combination in itself is novel, but we further improve it by introducing the ideas of 1) modulating the background model learning rate based on scene activity, and 2) making color-based segmentation criteria dependent on depth observations. Our experiments show that the method possesses much greater robustness to problematic phenomena than the prior state-of-the-art, without sacrificing real-time performance, making it well-suited for a wide range of practical applications in video event detection and recognition.

1. Introduction

Most systems that attempt to detect and recognize events in live video, and particularly such systems in the areas of surveillance and vision-based human-computer interaction, rely heavily on an early step, commonly called "background subtraction", that segments the scene into novel ("foreground") and non-novel ("background") regions. While improvement of the succeeding analysis steps in these systems has been the focus of a great deal of recent work, the systems' overall reliability usually depends as much, if not more, on the accuracy and robustness of their background subtraction methods.

Despite its importance, background subtraction remains a poorly solved problem. This is painfully apparent to the increasing number of scientists and engineers who have attempted to bring computer vision applications to the marketplace. While their methods for background subtraction may have been adequate for controlled laboratory settings and experimental demonstrations, they often fail catastrophically when confronted with a variety of real-world phenomena that commonly occur in settings such as homes, offices, and retail stores. Among the most problematic are:

Illumination changes: The background appearance changes with the lighting, and such changes can be confused with the introduction of foreground objects.

Shadows and inter-reflections: Moving foreground objects can create local illumination changes on background parts of the scene.

Background object changes: When objects are added to or removed from the scene (e.g. when someone picks up or leaves a package), or when an object considered background is changed (e.g. when a chair is moved), we may want to modify the background model if the change persists for a long enough time.

Camouflage: Similarity between the appearance of a foreground object and the background makes it difficult to distinguish between them.

Non-static background: Objects like changing television displays, foliage in wind, or a fan rotating in front of a wall, are not easily modeled by simple pixel statistics.

High-traffic areas: If the true background is frequently obstructed by many different foreground objects, it can be difficult to determine what the true background is and to segment foreground from it.

Some of these problems can be handled by very computationally expensive methods, but in many applications, a short processing time is required. For instance, most vision-based human-computer interfaces must react in a timely manner to a person's requests in order to be useful, and most vision-based surveillance systems collect far too much video to analyze off-line at some later time. Background subtraction methods used in such systems, therefore, need to keep pace with the flow of video data, so that the systems maintain some degree of "real-time" responsiveness.

This paper describes an algorithm for foreground segmentation that is the first to exhibit robustness to all of the

Algorithm Characteristics					*Example*	*Robustness to Various Phenomena*										
Color Input	Depth Input	Time Adaptive	Luminance Norm'ed Color	Multimodal BG Stats		Low Visual Texture	Depth Discontinuities	Non-Empty Scene Init	Illumination Changes	BG Object Changes	Shadows & Inter-Reflect	Color Camouflage	Depth Camouflage	Rotating Fan	Active Display	High-Traffic Areas
√						√	√						√			
√		√				√	√	√	√	√			√			
√		√	√		Pfinder[9]	√	√	√	√	√	√		√			
√		√		√	Stauffer et.al.[7]	√	√	√	√	√			√	√		
	√	√			Eveland et.al.[1]			√	√	√	√	√			√	√
√	√		√		Gordon et.al.[3]	√	√				√	√	√		√	√
√	√	√	√	√	this method	√	√	√	√	√	√	√	√	√	√	√

Table 1. Summary of robustness of several background subtraction methods to a variety of common problems. Methods given a √ for a particular problem do not all handle it equally well, but at least contain some feature designed to address it.

problems described above without sacrificing real-time performance. We achieve this in large part by taking advantage of the recent development of real-time depth computation from stereo cameras [5, 6, 8]. To first-order, our method can be summarized as an extension of the time-adaptive, per-pixel, Gaussian mixture modeling of color video, as described recently by Stauffer and Grimson [7], to the combined input space of color and depth imagery, as explored by Gordon, et. al. [3]. We seek to combine the best aspects of both methods, and we also introduce interesting new concepts to further improve the method's performance. The resulting robustness is significantly greater than that of prior algorithms, and yet, provided that real-time depth imagery is available[1], our method can still run in real-time on standard hardware.

2. Previous work

Many approaches to background subtraction have been proposed over the last few decades. The simplest class of methods, often found in experimental computer vision systems, use color or intensity as input, and employ a background model that represents the expected background feature values at each pixel as an independent (in image space), static (in time), unimodal (i.e. single-peaked, in feature space) distribution. For example, the background statistics are often modeled at each pixel using a single Gaussian, stored as a color mean and covariance matrix. The model is built during a "learning" phase while the scene is known to be empty of foreground objects, and is never modified thereafter. Typically, foreground is detected on a per-pixel basis wherever the current input image differs significantly from the distribution of expected background values. While such methods are well-suited for fast implementation and run time, they fail when confronted with any of the common, real-world phenomena listed in section 1.

Many methods have attempted to address some, but not all, of these problems, by improving one or more aspects of the basic class of methods. For example, by allowing the background model statistics to be updated over time, a system can adapt to gradual illumination changes, does not require an "empty" scene for initialization, and can incorporate persistent additions, removals, or alterations of objects into its background model. Furthermore, if color comparisons are made in a luminance-normalized space, some robustness to shadows and inter-reflections is obtained. Both of these ideas are incorporated in [9], among other systems. By changing the per-pixel models of expected background features from unimodal probability densities, such as single Gaussians, to more sophisticated representations, such as Gaussian mixture models, as in [2, 7], one can begin to address complex, non-static backgrounds. All of these methods still have great difficulty with color camouflage, active video displays, and high-traffic areas, among other things.

Others have taken advantage of the recent development of real-time depth computation from stereo cameras. Because the shapes of scene objects are not affected by illumination, shadows, or inter-reflections, depth information provides much robustness to such phenomena. Depth data is unreliable, however, in scene locations with little visual texture or that are not visible to all cameras, and tends to be noisy even where it is available. Hence, background subtraction methods based on depth alone [1, 4] produce unreliable answers in substantial portions of the scene, and often fail to find foreground objects in close proximity to the background, such as hands placed on walls or feet on a floor. A method using both depth and color has been proposed recently [3], but it lacks time adaptivity and uses a relatively simple statistical model of the background.

By combining the best of all of these ideas, our algorithm is able to perform reasonably well under all of the conditions listed in section 1. Table 1 summarizes the characteristics and capabilities of the algorithms described above, and compares them against our method. Note that the union of the rows for methods [7] and [3] produces the row corresponding to our method. However, due to further innovations that we introduce, such as activity-based modulation of model adaptation, we believe that our method handles several problem cases significantly better than [7], [3], and all other classes of methods in Table 1.

3. Method

Our method uses an approximation to Gaussian mixture modeling to describe the recent history of color and depth scene observations at each pixel. The models are updated

[1]Usage of hardware and software for real-time dense depth computation is growing quickly as costs decline rapidly. At least one vendor plans to provide this functionality in the near future at prices around US$100.

as new observations are obtained, while older data becomes less influential. Since these models may include observations of both foreground and background, we choose at each time step a subset of the Gaussians in each mixture as the current model of background appearance. The current observation is classified as background if it matches that part of the model, and as foreground otherwise. We discuss this basic method, and several enhancements, in detail below. An overview of the method is shown in Figure 1.

3.1. Gaussian mixtures in color and depth

The input to the algorithm is a time series of spatially-registered, time-synchronized pairs of color and depth images obtained by static cameras. Each pair of corresponding pixels in the two images for a given time step provides one scene observation in the combined input space of color and depth. We represent color in the YUV space, which separates luminance and chroma. We use depth, denoted as D, instead of disparity, so that our algorithm is applicable in systems that compute depth not only by image area-matching techniques, but also by methods based on active illumination, lidar, or other means. The observation at pixel i at time t can be written as $\vec{X}_{i,t} = [\ Y_{i,t} \quad U_{i,t} \quad V_{i,t} \quad D_{i,t}\ ]$.

The recent history of observations at a given pixel, $[\vec{X}_{i,1}, \ldots, \vec{X}_{i,t-1}]$, is regarded as a statistical process independent of that for all other pixels, and is modeled by a mixture of K Gaussian distributions. This is also the basis of the methods of [2, 7], except that they use an observation space of only non-luminance-invariant (RGB) color. The probability of the current observation at pixel i, given the model built from observations until the prior time step, can then be estimated as

$$P(\vec{X}_{i,t}|\vec{X}_{i,1}, \ldots, \vec{X}_{i,t-1}) = \sum_{k=1}^{K} w_{i,t-1,k} * \eta(\vec{X}_{i,t}, \vec{\mu}_{i,t-1,k}, \Sigma_{i,t-1,k}) \quad (1)$$

where η is a Gaussian probability density function, where $w_{i,t-1,k}$ is the weight associated with the k^{th} Gaussian in the mixture at time $t-1$, , and where $\vec{\mu}_{i,t-1,k}$ and $\Sigma_{i,t-1,k}$ are the mean $YUVD$ vector and covariance matrix of this k^{th} Gaussian. We denote the kth Gaussian distribution of a mixture as η_k. The weights $w_{i,t-1,k}$ indicate the relative proportions of past observations modeled by each Gaussian.

Rather than use full covariance matrices, we assume that all observation component measurements are independent, and that the chroma components have the same variance:

$$\Sigma = \mathrm{diag}[\ \sigma_Y^2 \quad \sigma_C^2 \quad \sigma_C^2 \quad \sigma_D^2\ ] \quad (2)$$

σ_Y^2, σ_C^2, and σ_D^2 are the luminance, chroma, and depth variances, respectively.

We choose K to be the same for all pixels, typically in the range of 3 to 5. As more Gaussians are used, the model is better able to represent more complex scenes, but at the cost of slowing the algorithm.

3.2. Invalid depth and chroma

Our methods for constructing and updating these Gaussian mixture models are complicated by the fact that many color and depth observations are not reliable. Therefore, before describing these methods, we discuss how we detect and represent "invalid" measurements.

The chroma (U and V) components of our color representation become unstable when the luminance is low. We therefore define a chroma validity test, based on luminance, as $\mathrm{CVal}(Y) \equiv Y > Y_{min}$. We also define this predicate to operate on a model Gaussian η_k, by applying it to the luminance mean of the distribution: $\mathrm{CVal}(\eta_k) \equiv \mu_{Y,k} > Y_{min}$. When $\mathrm{CVal}(Y)$ or $\mathrm{CVal}(\eta_k)$ are false, we do not use the chroma components of the current observation or the Gaussian distribution.

Stereo depth computation relies on finding small area correspondences between image pairs, and therefore does not produce reliable results in regions of little visual texture and in regions, often near depth discontinuities in the scene, that are visible in one image but not the other. Most stereo depth implementations attempt to detect such cases and label them with one or more special values, which we denote collectively as *invalid*. We rely on the stereo depth system to detect *invalid* depth, and we apply no extra tests. We define the following simple depth validity predicate: $\mathrm{DVal}(D) \equiv (D \neq \mathit{invalid})$.

Imager noise and subtle lighting variability can cause the depth measurement at a pixel to flicker in and out of validity, even if nothing is actually changing in the scene. Also, shadows often provide the texture needed to extract valid depth in regions where measurements are usually *invalid*. For these reasons, our background subtraction algorithm allows the same Gaussian η_k to represent a set of observations that contain both valid and *invalid* depth measurements. If many of these measurements are *invalid*, however, we regard the depth mean and variance of η_k as unreliable, and we do not use them in comparisons with the current observation. Specifically, we use the depth statistics of η_k only if, among the observations represented by η_k, the fraction with valid depth exceeds some threshold. We extend the predicate DVal to operate on Gaussian distributions accordingly:

$$\mathrm{DVal}(\eta_k) \equiv \frac{v_k}{w_k} > \rho \quad (3)$$

w_k is representative of the total number of observations that contributed to η_k, and v_k indicates the number of these observations that had valid depth. Methods for maintaining these variables over time are given in section 3.3. In our current system, we set ρ to a relatively low value, such as 0.2, so that we are able to estimate the depth of a scene location even when the depth measurement system cannot extract a reliable value there most of the time.

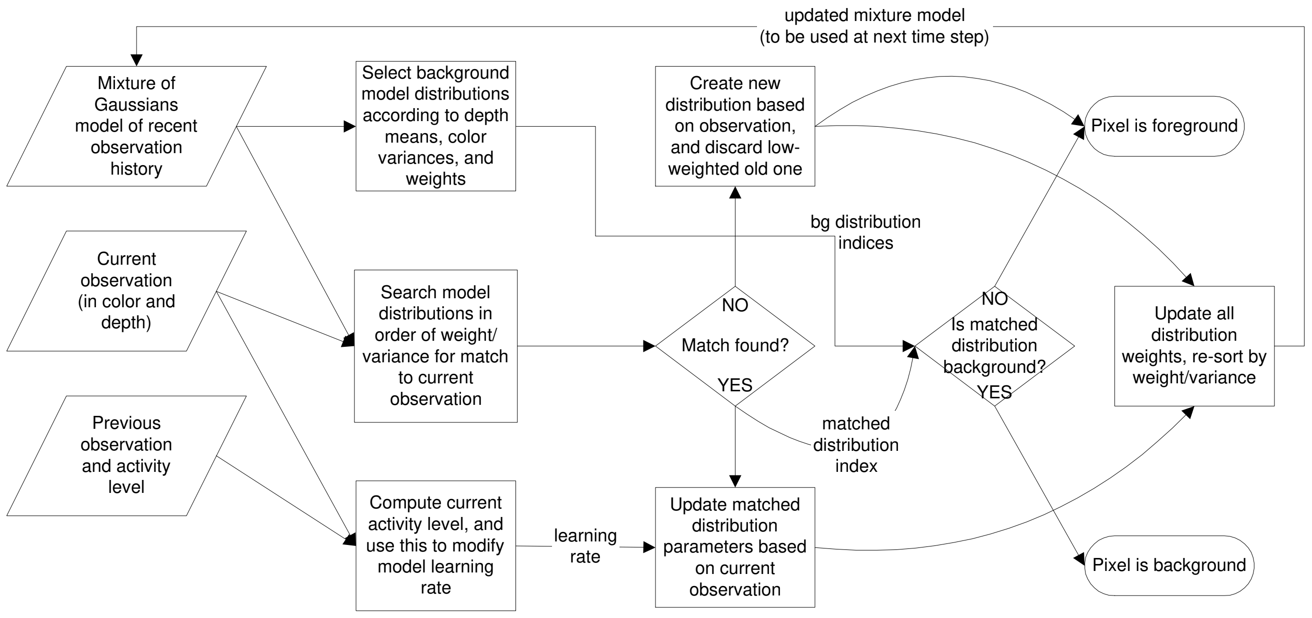
updated mixture model
(to be used at next time step)
Mixture of Gaussians model of recent observation history
Select background model distributions according to depth means, color variances, and weights
Create new distribution based on observation, and discard low-weighted old one
Pixel is foreground
bg distribution indices
Current observation (in color and depth)
Search model distributions in order of weight/ variance for match to current observation
NO
Match found?
YES
NO
Is matched distribution background?
YES
Update all distribution weights, re-sort by weight/variance
matched distribution index
Previous observation and activity level
Compute current activity level, and use this to modify model learning rate
learning rate
Update matched distribution parameters based on current observation
Pixel is background

Figure 1. Overview of per-pixel model adaptation and foreground/background decision processes. This diagram shows what occurs at a single pixel when a new input observation is processed.

3.3. On-line clustering of observations

The Gaussian mixtures used to model the history of scene observations at each pixel must be updated as new observations are obtained. Ideally, at each time step t, we would re-estimate the parameters for a pixel's mixture by applying an exact Expectation-Maximization algorithm on some time window of recent observations that includes the latest one. Unfortunately, this is a very costly procedure, so we instead use an on-line K-means approximation similar to that of [7]. The basic idea is to attempt to match the current observation $\vec{X}_{i,t}$ at a given pixel with one of the Gaussians η_k for that pixel, and, if a match is found, to adapt the parameters of η_k using the current observation. If no match is found, we replace one of the existing η_k with a new one that represents the current observation.

One way to carry out the matching process is to compute a "matching score" between $\vec{X}_{i,t}$ and each η_k in the pixel's mixture model, and then to choose as a match the η_k that maximizes this score, provided it is above some threshold. A good choice for a matching score is the weighted Gaussian probability density evaluated at $\vec{X}_{i,t}$. The matching distribution k would be selected as

$$k = \arg\max_k (w_{i,t-1,k} * \eta(\vec{X}_{i,t}, \vec{\mu}_{i,t-1,k}, \Sigma_{i,t-1,k})) \quad (4)$$

This matching score favors η_k with high weight and low variance: if $\vec{X}_{i,t}$ is similarly close to the means of two η_k, the one with higher weight and lower variance will likely have the higher score. Such η_k often model observations corresponding primarily to the true scene background. This is because the background consists usually of persistent, and often static, objects and processes. Therefore, η_k that largely model background are likely to account for more of the weight of recent observations, and often have small variances that reflect the unchanging themes of the background. The above matching procedure hence favors the formation of inclusive, yet compact, models of the background.

Because this method is very computationally expensive, we instead use an approximation that sacrifices little accuracy, continues to promote compact background descriptions, and has a runtime that is, in practice, no longer linear in K. Rather than compute the matching score for all η_k, we first sort the η_k in order of decreasing weight/variance, and then select as the best match the first Gaussian whose mean is sufficiently near to $\vec{X}_{i,t}$. The dependence of (4) on the weight and variance of η_k are addressed by the sorting, while the dependence on the mean of η_k is factored into the mean comparison. We use only the luminance variance, rather than the norm of the full covariance matrix, $|\Sigma_{i,t-1,k}|$, for the sorting, because of the possibility of unreliable depth or chroma data, and because of the higher noise levels typical of depth measurements.

The matching criterion $\mathcal{M}$ we use is separated into a depth-based portion $\mathcal{M}_D$ and a color-based portion $\mathcal{M}_C$, both of which must be true for a match to occur. When the color and depth of both $\vec{X}_{i,t}$ and η_k are valid, we require that all squared differences between corresponding components of $\vec{X}_{i,t}$ and the mean of η_k are less than some small multiple β of η_k's corresponding component variance. In the absence of valid depth data in either $\vec{X}_{i,t}$ or η_k, $\mathcal{M}_D$

assumes a match. Similarly, in the absence of valid chroma data in either $\vec{X}_{i,t}$ or η_k, $\mathcal{M}_C$ uses only luminance to make a decision. The full matching criterion, omitting the i, t, and k subscripts, can be written as:

$$\begin{aligned}
\mathcal{M} &\equiv \mathcal{M}_D \wedge \mathcal{M}_C, \text{ where} \qquad (5)\\
\mathcal{M}_D &\equiv \neg\mathrm{DVal}(D) \vee \neg\mathrm{DVal}(\eta_k) \vee ((D-\mu_D)^2 < \beta^2\sigma_D^2)\\
\mathcal{M}_C &\equiv [\mathrm{CVal}(Y) \wedge \mathrm{CVal}(\eta_k) \wedge ((Y-\mu_Y)^2 < \beta^2\sigma_Y^2) \wedge\\
&\quad ((U-\mu_U)^2 + (V-\mu_V)^2 < \beta^2\sigma_C^2)] \vee\\
&\quad [\neg(\mathrm{CVal}(Y) \wedge \mathrm{CVal}(\eta_k)) \wedge ((Y-\mu_Y)^2 > \beta^2\sigma_Y^2)]
\end{aligned}$$

The parameter β is typically chosen to be about 2.5 to 3, so that the boundary of the matching zone in YUVD-space for η_k encompasses over 90% of the data points that would be drawn from the true Gaussian probability density.

If $\vec{X}_{i,t}$ and η_k are found to match, we incrementally adapt the parameters of η_k toward $\vec{X}_{i,t}$ by applying the following equations:

$$\begin{aligned}
\vec{\mu}_{i,t,k} &= (1-\alpha)\vec{\mu}_{i,t-1,k} + \alpha\vec{X}_{i,t}\\
\sigma^2_{Y,i,t,k} &= (1-\alpha)\sigma^2_{Y,i,t-1,k} + \alpha(Y_{i,t} - \mu_{Y,i,t-1,k})^2\\
\sigma^2_{C,i,t,k} &= (1-\alpha)\sigma^2_{C,i,t-1,k} + \qquad (6)\\
&\quad \alpha(\vec{X}_{C,i,t} - \vec{\mu}_{C,i,t-1,k})^T(\vec{X}_{C,i,t} - \vec{\mu}_{C,i,t-1,k})\\
\sigma^2_{D,i,t,k} &= (1-\alpha)\sigma^2_{D,i,t-1,k} + \alpha(D_{i,t} - \mu_{D,i,t-1,k})^2\\
v_{i,t,k} &= (1-\alpha)v_{i,t-1,k} + \alpha(\mathrm{DVal}(D))
\end{aligned}$$

$\vec{X}_C$ and $\vec{\mu}_C$ denote the chroma subvectors of the pixel observation and the mean of η_k, and $v_{i,t,k}$ is the counter of valid depth observations discussed in section 3.2. The parameter α can be interpreted as a learning rate: as α is made smaller, the parameters of η_k will be perturbed toward new observations in smaller incremental steps. Also, $1/\alpha$ effectively determines the duration of the time window of "recent" observations that the Gaussian mixture models represent, with values further in the past being weighted in an exponentially decreasing manner.

All variances are not allowed to decrease below some minimum value, so that matching does not become unstable in scene regions that are static for long time periods. Also, the luminance variance floor was kept at a substantial level, so that luminance changes caused by shadows and inter-reflections would less often result in matching failures.

The weights for all Gaussians are updated according to

$$w_{i,t,k} = (1-\alpha)w_{i,t-1,k} + \alpha\mathcal{M}_{i,t,k} \qquad (7)$$

$\mathcal{M}_{i,t,k}$ is 1 (true) for the η_k that matched the observation and 0 (false) for all others, so (7) causes the weight of the matched η_k to increase and all other weights to decay.

If no match is found, the Gaussian ranked last in weight/variance is replaced by a new one with a mean equal to $\vec{X}_{i,t}$, an initially high variance, a low initial weight (α), and $v_{i,t,k} = \alpha\mathrm{DVal}(D)$.

3.4. Background model estimation

At each time step, one or more of the Gaussians in each per-pixel mixture are selected as the background model, while any others are taken to represent foreground. We designate the current observation at a pixel to be part of the foreground if it was not matched to any of the η_k in the pixel's current background model.

We select background Gaussians at a given pixel according to two main criteria: **1)** select a first, "deepest" Gaussian, if possible, and **2)** select additional Gaussians, if necessary, until their total weight is sufficiently high. The first criterion is based on the fact that, in general, we do not expect to be able to see through the background. The second is based on the notion that the background model should account for a reasonably high fraction of past observations. We discuss each criterion in more detail below.

For the first criterion, we choose the Gaussian η_k with the largest depth mean, but we consider only those Gaussians for which $\mathrm{DVal}(\eta_k) = \mathit{true}$ and whose weight is above a threshold T_D. These restrictions discourage the selection of a deepest background η_k based on depth measurement noise or transient observations. They also help us handle image regions where the true background seldom, if ever, produces reliable depth measurements, perhaps because of low visual texture. In such cases, the η_k representing the true background has *invalid* depth statistics, while the Gaussian with the largest valid depth mean might be one with low weight corresponding to a seldom-observed foreground object. The weight threshold T_D helps us avoid selecting the latter Gaussian as background. We typically set T_D around 0.1-0.2, so that we can select an η_k representing the true background even when it is usually not visible.

For the second criterion, we add Gaussians in order of decreasing weight/variance to the background model until the total weight of all background Gaussians exceeds a threshold T. This favors Gaussians that model frequent, consistent observations. The second criterion is useful where the true background usually produces *invalid* depth data, so that our first criterion fails to find any Gaussian with valid depth and significant weight. It is also useful where the background is truly multimodal, such as for a fan rotating in front of a wall, or trees waving in the wind. We need to use multiple η_k to model the background well in such cases, and we want to select enough to account for a significant fraction of recent observations.

3.5. Linking depth and color matching

In the method for matching an observation $\vec{X}_{i,t}$ to a Gaussian η_k described in section 3.3, color and depth are analyzed independently, so that a failure to match in either of these spaces causes the overall match to fail. Hence, if the difference between $\vec{X}_{i,t}$ and η_k in one space is just outside the acceptable tolerance, the overall match will be rejected even if evidence in the other space strongly supports it. It would seem, intuitively, that we could make this process less brittle, and improve our model's integrity, by introducing some linkage between comparisons in color and depth.

We achieve this by allowing the depth match decision for a particular $\vec{X}_{i,t}$ and η_k to bias the companion color match decision in the same direction. This is done for two cases of depth matching, and in each case, the bias is implemented by adjusting the parameter β before the color matching test $\mathcal{M}_C$ is applied. The first case occurs when both $\vec{X}_{i,t}$ and η_k have valid depth and are near each other, thereby satifying the last predicate of $\mathcal{M}_D$. In such instances, where depth measurements give a positive indication of a match, we increase β, typically by a factor of 2, so that we are more lenient in our color matching. This idea, discussed in more detail in [3], helps mitigate the tradeoff in setting β high enough to tolerate appearance changes due to shadows, inter-reflections, and lighting changes, but low enough to avoid foreground omissions due to color camouflage (similar color of foreground and background). By increasing β only where depth indicates a probable match, we are better able to ignore strong shadows in such cases, while not compromising the restrictiveness of our color matching within regions of uncertain depth. This is also very helpful in modeling dynamic background objects, such as video displays or distant trees waving in the wind, that have somewhat constant depth characteristics but highly varying color. In these instances, we rely more heavily on depth comparisons and reduce our sensitivity to large fluctuations in color.

The second case in which we allow depth data to influence color matching occurs when the validity of the current depth observation does not match that of η_k. This commonly happens when a foreground object with valid depth passes in front of a background region with low visual texture that produces primarily *invalid* depth. It also happens frequently in the reverse situation, when the background depth is valid but that of the foreground is not, often due to the depth discontinuity between foreground and background. Most methods that use depth do not allow matching between valid and *invalid* depth, and will therefore handle the above cases well. However, because changes in depth validity at a scene location are frequently caused by shadows, imager noise, and lighting variability, these methods tend to produce a very noisy foreground segmentation. The predicate $\mathcal{M}_D$ we give in (5), on the other hand, *always* allows matching when there is a mismatch in depth validity, thereby relying on color alone to make foreground decisions in such cases. Neither of these options is entirely satisfactory, and we would like to find a middle ground.

We do this by regarding a mismatch in depth validity between $\vec{X}_{i,t}$ and η_k as a likely, but not definite, indicator that the overall match should be rejected, and we therefore tighten the color matching threshold. Specifically, when $\text{DVal}(D) \neq \text{DVal}(\eta_k)$, we decrease β, typically by a factor of 2, before the color matching criterion $\mathcal{M}_C$ is applied. This dramatically reduces foreground omissions due to color camouflage in regions where either the foreground or background depth is unreliable because of low visual texture or depth discontinuities.

3.6. Activity-based learning modulation

As the learning rate α is increased, static changes to the background, such as the moving of a chair, are correctly incorporated into the background model more quickly. Unfortunately, the model of the true background is also lost more rapidly in regions of high foreground traffic, and true foreground objects that remain relatively static, such as two people who have stopped to have a conversation, fade more quickly into the background. We mitigate this tradeoff by exploiting the tendency for the scene "activity" level to be higher in the latter two cases than in the first. Specifically, we compute a scene activity measure A at each pixel, and we reduce the learning rate α (used in the update equations (6) and (7)) by some factor ξ at all pixels where A exceeds a threshold H. A moved, but now static, chair will then continue to be incorporated into the background model quickly, while foreground objects that are not completely motionless will be incorporated less quickly.

The activity level A at each pixel is set to zero for the first time step, and thereafter is computed recursively from the previous activity level and the luminance difference between the current frame and the last:

$$A_{i,t,k} = (1-\lambda)A_{i,t-1,k} + \lambda\,|Y_{i,t} - Y_{i,t-1}| \qquad (8)$$

The activity learning rate λ helps temporally smooth the frame differences, and helps to keep the activity level high within the interior of low-texture objects as they move across the scene. The threshold H is set as low as possible without causing imager noise to drive A above H at a substantial fraction of pixels. Depth is not used in computing A in our system because of the higher depth noise level. We typically set ξ in the range of 5 to 20; the higher the value of ξ, the longer non-static foreground objects may obscure the background before being incorporated into it. α is not reduced to zero, so that our method is able to learn models of dynamic backgrounds, albeit more slowly.

4. Experimental results

We have implemented our algorithm to run in real-time on a standard PC platform. With little optimization, the method runs at 8Hz on 320x240 images on a 500MHz Pentium. Color and depth input were supplied by a stereo camera head with special-purpose hardware for depth computation, based on [8]. To test our algorithm in a reproducible manner, however, we used file sequences captured from the stereo camera pair at 320x240 resolution and 15Hz. In this section, we compare the performance of our method on a challenging test sequence to several alternatives.

We denote our method as (A), and we compare it against several methods in which a single key component of (A) has been removed. Specifically, we tried (B) removing depth information, (C) removing color information, (D) using an RGB color space instead of YUV, (E) decreasing the per-pixel background model complexity from Gaussian mixtures to single Gaussians, (F) removing activity-based learning modulation, and (G) removing depth-based modulation of color matching. Methods (B), (C), and (E) are related to [7], [1], and [3], respectively, but in each case, the experimental method contains additional features, such as activity-based learning rate modulation, that significantly improves its performance beyond that of the related prior work. For all methods, the learning rate α was set to produce an adaptation time window of several minutes.

The test sequence is 10 minutes long, with no image being devoid of "foreground" people. It contains several dynamic background objects, namely several video displays (toward the upper left of the images) and a sign rotating about a vertical axis at about 0.5Hz (upper middle of images, sitting on oval-shaped table). During the first half of the sequence, two displays ("display1" and "display2") are active and one ("display3") is off, while two people walk around the room. Near the midpoint of the sequence, the chair in the lower left of the image is moved to new floor position, "display2" is switched off, "display3" is switched on, and several more people enter the scene. One of these people stands near the middle of the back of the room, and remains there for the rest of the sequence, occasionally shifting his weight or moving his arms. The other new people walk around continuously in the lower right portion of the image, creating a "high-traffic" area.

For this sequence, the ideal foreground segmentation would, in all frames, contain nothing but the people and any objects they manipulate. The video displays and the rotating sign would always be modeled as part of the background. Foreground objects would be segmented without "holes" (missing pieces) and without their shadows. However, for any pixel-level segmentation method with no input from higher level modules, we also expect some temporary errors while the method adapts to changes made to the true background. For instance, when the chair is moved, we expect to see, for some amount of time, two erroneous foreground regions corresponding to the new and old locations of the chair. Also, we expect temporary, undesirable foreground inclusions near the beginning of the sequence, while the method learns models for parts of the scene that were initially occluded by foreground objects. We would like to minimize the occurrence and duration of such errors.

Our experiments indicated that our method came much closer to achieving this ideal segmentation than any of the alternatives. We discuss our general observations below, and will refer to the selected, illustrative frames shown in Figure 2. All images in each column of the figure correspond to the same frame from each sequence; the top row shows the original video frame luminance, while the remaining rows show the foreground extracted by the various methods. Several of the test sequence challenges described above are indicated in result images for method (B).

In general, our method extracted people more consistently and with many fewer holes than any of the alternatives. We observe fewer holes in (A) due to color or depth camouflage than in (B) and (C), since (A) has an additional measurement source to disambiguate each of these situations. Very few shadows or inter-reflections were observed in the foreground for our method, largely as a result of our use of a luminance-invariant color space, as can be seen from comparing frames of methods (A) and (D). Also, our method rarely included the active video displays in the foreground, and usually did not include the rotating sign. For the sign in particular, it is evident that this success was due in part to the use of multimodal (mixture of Gaussian) background models, since the sign appears in several result frames for the unimodal method (E). However, for the sign and particularly for the active video displays, the loosening of the color matching criterion in the presence of depth matches was also important. This is evidenced by the more frequent appearance of the active displays in results for (G), which did not use depth-based modulation of color matching, and for (B), which did not use depth information at all.

In comparing the results of (B) and (G) to those for (A), we also see fewer holes in the foreground people in the high-traffic area. This can be explained by first noting that the true scene background here contains a large amount of *invalid* depth, while the people usually have valid depth data. In addition, several people in the scene are wearing blue jeans, while the carpet is also blue. Method (A) tightens the color matching criterion when the validity of the current observation and background model do not match, and therefore produces fewer foreground omissions due to color camouflage in this part of the scene. Methods (B) and (G) do not do this, and have greater camouflage problems.

Activity-based learning rate modulation proved very helpful in minimizing temporary errors due to background changes, without compromising robustness to other phenomena. For example, the change in the location of the chair was incorporated into the background model by (A) in less than 2 minutes, while (A) segmented people accurately in the high-traffic area for the full 5 minutes of its duration. The slower background model learning rate in the presence of activity largely accounts for this time difference, as can be seen from the more significant foreground omissions in the high-traffic area in the late frame results of (F).

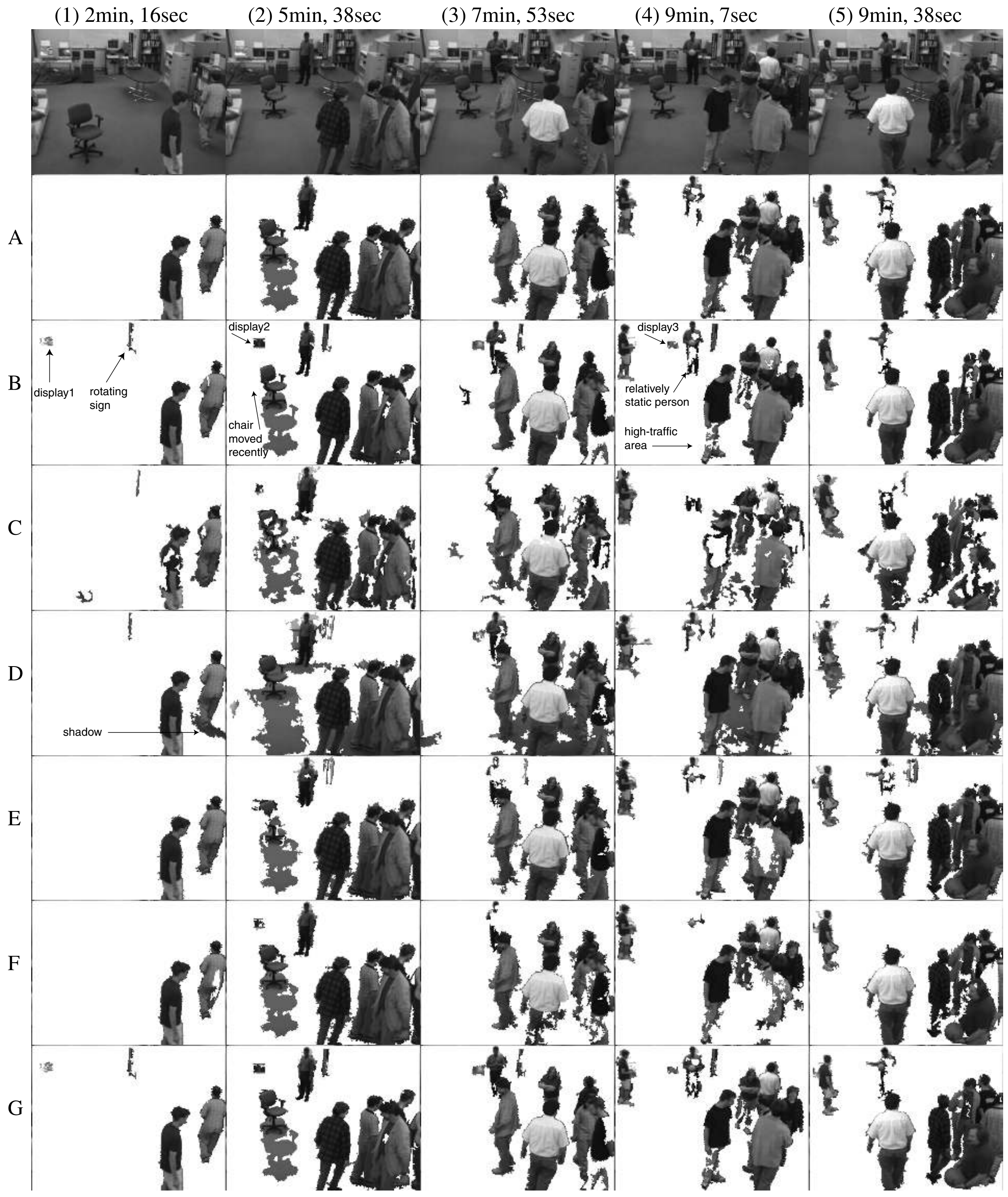

Figure 2. Comparison of our method to close relatives, at selected frames of a challenging video sequence. Color input is shown in top row; input depth is not shown. Result rows are for the following methods: (A) the method of this paper; (B) no depth (color-only); (C) no color (depth-only); (D) RGB (instead of YUV); (E) unimodal background models (not multiple Gaussians); (F) no activity-based learning modulation; (G) no depth-based modulation of color matching. Where applicable, parameter settings were $K = 4, \alpha = 0.0006, \beta = 2.5, \rho = 0.2, Y_{min} = 16, T = 0.4, T_D = 0.2, \lambda = 0.09, H = 5, \xi = 5$. Small, isolated foreground regions, and small holes within foreground, were removed by applying an area threshold to the results of connected-components analysis.

Activity-based learning modulation was also the most important factor in enabling (A) to segment a significant portion of the relatively inactive person in the upper-middle of the later frames. Although this person did not move much, he moved enough to cause the learning rate to be lowered at his location much of the time, and he was almost always segmented, at least in part, for the entire 5 that minutes he stood at his position. In contrast, in method (F), where no learning rate modulation was used, he was almost entirely incorporated into the background model after just 1.4 minutes, so that only small parts of him occasionally appeared in the foreground thereafter.

For (C), in which color is not available, we improved results by modeling valid and *invalid* depth with separate Gaussians, as in [1], rather than combine them in single Gaussians as described in section 3.2. Despite this, results for (C) were rather poor, largely because of the substantial low-texture image regions, such as the floor, that produce very noisy, often *invalid* depth data. At many pixels, the floor is modeled primarily by one Gaussian for which $\mathrm{DVal}(\eta_k) = false$. Foreground objects passing through these pixels, on the other hand, tend to have valid depth and are modeled by Gaussians with $\mathrm{DVal}(\eta_k) = true$. According to the criteria for selecting background model Gaussians discussed in section 3.4, the latter Gaussians will be chosen preferentially, and incorrectly, when their weight exceeds T_D. This happens most in the high-traffic area, since the floor is frequently obscured here. The background model here quickly becomes, roughly, the outer hull of the space in which the people walk.

It is also important to note that, unlike in method (A), the depth-only method (C) is unable to distinguish different people passing through a pixel at similar depths, even if they have different color characteristics. This causes the observations representing people in the high-traffic area to be lumped into fewer Gaussians, whose weights rapidly grow and cause them to even more quickly dominate the Gaussians modeling the floor. Similar things occur in the color-only method (B): for instance, a person in a light gray shirt walking five feet in front of a white file cabinet may match the cabinet in color. Not only will the segmented person contain holes there, but he also will corrupt the model of the cabinet, making its mean a little more gray as his color data is used to adapt it. Method (A), on the other hand, uses different Gaussians because of the very different depths of these objects, and the colors associated with each Gaussian more accurately reflect the respective objects. These examples point out a substantial, if subtle, advantage of using combined input space of depth and color: observations corresponding to different objects may overlap substantially in either space alone, but often do not in the combined space. Methods that use both color and depth are therefore able to build statistically cleaner models of the background, and hence tend to produce cleaner foreground segmentations.

5. Conclusions

We have described a new algorithm for background estimation and removal in video that possesses much greater robustness than prior state-of-the-art to a variety of common, problematic, real-world phenomena. Because our method operates only at the pixel level, it is well-suited for combination with any of a wide variety of foreground segmentation techniques that consider inter-pixel relationships, analyze regional scene properties, or use high-level input from modules such as object trackers or identifiers. While it operates in real-time on standard hardware, the fact that it processes each pixel independently also makes it highly amenable to a much faster, parallel implementation. This combination of speed and reliability creates a solid background subtraction platform on which to build a wide variety of surveillance, human-computer interaction, and other applications for practical, relatively unconstrained settings.

References

[1] C. Eveland, K. Konolige, R. Bolles. "Background Modeling for Segmentation of Video-rate Stereo Sequences". In *CVPR'98*, pp.266-271, June 1998.

[2] N. Friedman, S. Russell. "Image Segmentation in Video Sequences: a Probabilistic Approach". In *13th Conf. on Uncertainty in Artificial Intelligence*, August 1997.

[3] G. Gordon, T. Darrell, M. Harville, J. Woodfill. "Background Estimation and Removal Based on Range and Color". In *CVPR'99*, Vol.2, pp.459-464, June 1999.

[4] Y. Ivanov, A. Bobick, J. Liu. "Fast Lighting Independent Background Subtraction". *Int. J. Comp. Vis.*, 37(2), pp.199-207, June 2000.

[5] K. Konolige. "Small Vision Systems: Hardware and Implementation". *8th Int. Symp. on Robotics Research*, 1997.

[6] Point Grey Research, http://www.ptgrey.com

[7] C. Stauffer, W.E.L. Grimson. "Adaptive Background Mixture Models for Real-Time Tracking". In *CVPR'99*, Vol.2, pp.246-252, June 1999.

[8] J. Woodfill, B. Von Herzen. "Real-Time Stereo Vision on the PARTS Reconfigurable Computer". *Symposium on Field-Programmable Custom Computing Machines*, April 1997.

[9] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland. "Pfinder: Real-time Tracking of the Human Body". *IEEE Trans. on Patt. Anal. Mach. Intell.*, 19:7, July 1997.