Agenda

• BG subtraction
• HW 1
• Waiver forms (again!)
http://www.ics.uci.edu/~dramanan/teaching/cs117_spring11/

5 projects worth 20% each. Projects handed in 24 hours early will get 10% extra credit on the assignment (e.g. if the assignment is worth 100 points you will get 10 pts extra credit). Each project will require writing code in MATLAB, which is available on about 34 machines in the CS 364 lab - the machines are in 3 rows front of the lab assistant's desk and to the left of this desk as you face away from it.

Project 1 due in 1 week

We’ll go over necessary algorithms today
Background subtraction
Simple Background Subtraction

1) Assume we know B
2) How to efficiently implement in Matlab w/o for loops?
3) Grayscale vs color?
Let’s try the average frame (computed offline) as the background

- Background model is a static image (assumed to have no objects present).
- Pixels are labeled as object (1) or not object (0) based on thresholding the absolute intensity difference between current frame and background.

\[
B = \text{average}(I);
\]

\[
\ldots
\]

\[
\text{loop time } t
\]

\[
I(t) = \text{next frame};
\]

\[
\text{diff} = \text{abs}[B - I(t)];
\]

\[
M(t) = \text{threshold(diff}, \lambda); \\
\]

\[
\ldots
\]

\[
\text{end}
\]
Let’s try first frame as the background

- Background model is a static image (assumed to have no objects present).
- Pixels are labeled as object (1) or not object (0) based on thresholding the absolute intensity difference between current frame and background.

\[
B = I(0);
\]

\[
\text{loop time } t
\]

\[
I(t) = \text{next frame};
\]

\[
diff = \text{abs}[B - I(t)];
\]

\[
M(t) = \text{threshold}(\text{diff}, \lambda);
\]

\[
\text{end}
\]

Online algorithm
How well does it work?

Background subtraction does a reasonable job of extracting the shape of an object, provided the object intensity/color is sufficiently different from the background.
Objects that enter the scene and stop continue to be detected, making it difficult to detect new objects that pass in front of them.

If part of the assumed static background starts moving, both the object and its negative ghost (the revealed background) are detected.
More problems

Background subtraction is sensitive to changing illumination and unimportant movement of the background (for example, trees blowing in the wind, reflections of sunlight off of cars or water).

Background subtraction cannot handle movement of the camera.

How might this affect an actual interface?
Frame-differencing

- Background model is replaced with the previous image.

\[
\begin{align*}
B(0) &= I(0); \\
\ldots
\end{align*}
\]

\[
\begin{align*}
\text{loop time } t \\
I(t) &= \text{next frame;}
\end{align*}
\]

\[
\begin{align*}
diff &= \text{abs}[B(t-1) - I(t)]; \\
M(t) &= \text{threshold}(\text{diff}, \lambda);
\end{align*}
\]

\[
\begin{align*}
\ldots
\end{align*}
\]

\[
\begin{align*}
B(t) &= I(t);
\end{align*}
\]

end
How well does it work?

Frame differencing is very quick to adapt to changes in lighting or camera motion.

Objects that stop are no longer detected. Objects that start up do not leave behind ghosts.

However, frame differencing only detects the leading and trailing edge of a uniformly colored object. As a result very few pixels on the object are labeled, and it is very hard to detect an object moving towards or away from the camera.
Adjusting temporal scale of differencing

Note what happens when we adjust the temporal scale (frame rate) at which we perform two-frame differencing …

Define \( D(N) = \| I(t) - I(t+N) \| \)

more complete object silhouette, but two copies (one where object used to be, one where it is now).
A neat “trick”: 3-frame differencing

The previous observation is the motivation behind three-frame differencing.

D(-15)
where object was, and where it is now

AND

D(+15)
where object is now, and where it will be

where object is now!
But it's hard to find a good frame rate

Choice of good frame-rate for three-frame differencing depends on the size and speed of the object.

<table>
<thead>
<tr>
<th># frames skipped</th>
<th>1</th>
<th>5</th>
<th>15</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="#" alt="Frame 1" /></td>
<td><img src="#" alt="Frame 5" /></td>
<td><img src="#" alt="Frame 15" /></td>
<td><img src="#" alt="Frame 25" /></td>
</tr>
<tr>
<td></td>
<td><img src="#" alt="Frame 35" /></td>
<td><img src="#" alt="Frame 45" /></td>
<td><img src="#" alt="Frame 55" /></td>
<td><img src="#" alt="Frame 65" /></td>
</tr>
</tbody>
</table>

This worked well for the person.
Adaptive background subtraction

- Current image is "blended" into the background model with parameter $\alpha$
- $\alpha = 0$ yields simple background subtraction, $\alpha = 1$ yields frame differencing

\[ B(0) = I(0); \]

... loop time $t$

\[ I(t) = \text{next frame}; \]
\[ \text{diff} = \text{abs}[B(t-1) - I(t)]; \]
\[ M(t) = \text{threshold}(\text{diff}, \lambda); \]

... \[ B(t) = \alpha I(t) + (1-\alpha)B(t-1); \]
end
Use some previous method to identify foreground/background pixels

Mark each pixel with the last “time” it was declared foreground
Efficient implementation

- Motion images are combined with a linear decay term
- also known as motion history images (Davis and Bobick)

B(0) = I(0);
H(0) = 0;
loop time t
I(t) = next frame;
diff = abs[B(t-1) - I(t)];
M(t) = threshold(diff, λ);
tmp = max[H(t-1) - γ, 0];
H(t) = max[255 * M(t), tmp];
...
B(t) = I(t);
end
Deeper question

• What is the “right” way to do build a background model?

Next lecture: statistical color models
(Interlude) HW 1