Shadow Detection

Instructions: This tutorial will guide you how you can use the simple approaches for detecting shadows.

Prerequisites: You need to have OpenCV installed on your machine.

Installing OpenCV

If you do not have OpenCV installed on your machine yet, please complete the installation instructions first at `http://opencv.willowgarage.com/wiki/InstallGuide`.

Explaining the Folder Structure

Under the folder `workshop`, there are two folders: `code` and `data`. Under the folder `data`, there are two subfolders: `input` for storing the input data and `working` for storing the files generated by the program. Any file under the folder `working` can be deleted.

Under the folder `input`, there are four subfolders. The subfolders `bgs` containing background images, `frames` containing arbitrary frames, and `masks` containing mask images. These three subfolders will be used for training in the maximum likelihood approach. The subfolder `videos` containing three videos for testing the algorithms.

Experimenting with Shadow Detection Methods

In this section, we will experiment with three different shadow detection methods, namely, HSV-based thresholding method, normalized cross-correlation method, and maximum likelihood method.

HSV-based Thresholding Method

We use the HSV-based thresholding as a first method to detect shadows. The algorithm is defined as follows:

$$SP_t(x,y) = \begin{cases} 1 & \text{if } \alpha \leq \frac{I_t^V(x,y)}{B_t^V(x,y)} \leq \beta \\ & \quad \wedge (I_t^S(x,y) - B_t^S(x,y)) \leq T_S \\ & \quad \wedge \left| I_t^H(x,y) - B_t^H(x,y) \right| \leq T_H \\ 0 & \text{otherwise,} \end{cases}$$

where $SP_t(x,y)$ is the resulting binary mask for shadows at each pixel (x,y) at time t. I_t^H, I_t^S, I_t^V, B_t^H, B_t^S, and B_t^V are the H, S, and V components of foreground pixel $I_t(x,y)$ and background pixel $B_t(x,y)$ at pixel (x,y) at time t, respectively.

Look at the code `shadow_hsv.cc`. Try to understand how it works then run the program with the videos provided. Tune the parameters and see the results.

Normalized Cross-Correlation Method

In this section, we will experiment with the normalized cross-correlation (NCC) method.

For each pixel (i, j) of the foreground mask, it considers a $(2N + 1) \times (2N + 1)$ template T_{ij} defined by $T_{ij}(n, m) = I(i + n, j + m)$ for $-N \leq n \leq N$ and $-N \leq m \leq N$ (N is empirically determined). If $B(i, j)$ is the background model, the NCC value at pixel (i, j) is defined as follows:

$$NCC(i, j) = \frac{ER(i, j)}{E_B(i, j)E_{T_{ij}}},$$

where

$$ER(i, j) = \sum_{n=-N}^{N} \sum_{m=-N}^{N} B(i + n, j + m)T_{ij}(n, m),$$

$$E_B(i, j) = \sqrt{\sum_{n=-N}^{N} \sum_{m=-N}^{N} B(i + n, j + m)^2},$$

$$E_{T_{ij}} = \sqrt{\sum_{n=-N}^{N} \sum_{m=-N}^{N} T_{ij}(n, m)^2}.$$

For a pixel in a shadow region, the NCC value should be large (close to one) and the $E_{T_{ij}}$ for the region around (i, j), i.e., its magnitude, should be smaller than $E_B(i, j)$. Consequently, a pixel is classified as shadow if

$$NCC(i, j) \geq L_{NCC}$$

and

$$E_{T_{ij}} < E_B(i, j),$$

where L_{NCC} is an empirical threshold.

Look at the code `shadow_ncc.cc`. Try to understand how it works then run the program with the videos provided. Tune the parameters and see the results.

Maximum Likelihood Method

We divide this section into two phases. In the offline phase, we will extract the HSV color information and analyze the distributions over the difference in hue, saturation, and value components for foreground and shadow pixels. In the online phase, we will use the maximum likelihood (ML) approach to classify a pixel as a shadow pixel or an object pixel.

Offline Phase

In the offline phase, after foreground extraction, we manually label pixels as either shadow or object. We then observe the distribution over the difference in hue (H_{diff}), saturation (S_{diff}), and value (V_{diff}) components in the HSV color space between pixels in the current frame and the corresponding pixels in the background model.

We define the measurement likelihood for pixel (x, y) given its assignment as follows.

$$\begin{aligned} P(M_{\text{xy}} \mid A_{\text{xy}} = \text{sh}) \quad = \quad & P(H_{\text{diff}} \mid A_{\text{xy}} = \text{sh}) \\ & P(S_{\text{diff}} \mid A_{\text{xy}} = \text{sh}) \\ & P(V_{\text{diff}} \mid A_{\text{xy}} = \text{sh}), \end{aligned} \tag{1}$$

where M_{xy} is a tuple containing the HSV value for pixel (x, y) in the current image as well as the HSV value for pixel (x, y) in the background model for pixel (x, y), and A_{xy} is the assignment of pixel (x, y) as object

or shadow. "sh" stands for shadow.

To make the problem tractable, we assume that the distributions over the components on the right hand side in (1) follow Gaussian distributions, defined as follows.

$$P(H_{\text{diff}} \mid A_{\text{xy}} = \text{sh}) = \mathcal{N}(H_{\text{diff}}; \mu_{h_{\text{diff}}^{\text{sh}}}, \sigma^2_{h_{\text{diff}}^{\text{sh}}})$$

$$P(S_{\text{diff}} \mid A_{\text{xy}} = \text{sh}) = \mathcal{N}(S_{\text{diff}}; \mu_{s_{\text{diff}}^{\text{sh}}}, \sigma^2_{s_{\text{diff}}^{\text{sh}}})$$

$$P(V_{\text{diff}} \mid A_{\text{xy}} = \text{sh}) = \mathcal{N}(V_{\text{diff}}; \mu_{v_{\text{diff}}^{\text{sh}}}, \sigma^2_{v_{\text{diff}}^{\text{sh}}})$$

Similarly, the measurement likelihood for object pixels can be computed as follows.

$$\begin{aligned} P(M_{\text{xy}} \mid A_{\text{xy}} = \text{obj}) \quad = \quad & P(H_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) \\ & P(S_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) \\ & P(V_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) \end{aligned} \tag{2}$$

Here "obj" stands for object. As for the shadow pixel distributions, we assume Gaussian distributions over the components on the right hand side in (2), as follows.

$$P(H_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) = \mathcal{N}(H_{\text{diff}}; \mu_{h_{\text{diff}}^{\text{obj}}}, \sigma^2_{h_{\text{diff}}^{\text{obj}}})$$

$$P(S_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) = \mathcal{N}(S_{\text{diff}}; \mu_{s_{\text{diff}}^{\text{obj}}}, \sigma^2_{s_{\text{diff}}^{\text{obj}}})$$

$$P(V_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) = \mathcal{N}(V_{\text{diff}}; \mu_{v_{\text{diff}}^{\text{obj}}}, \sigma^2_{v_{\text{diff}}^{\text{obj}}})$$

We estimate the parameters $\Theta = \{\mu_{v_{\text{diff}}^{\text{sh}}}, \sigma^2_{v_{\text{diff}}^{\text{sh}}}, \mu_{v_{\text{diff}}^{\text{sh}}}, \sigma^2_{v_{\text{diff}}^{\text{sh}}}, \mu_{v_{\text{diff}}^{\text{sh}}}, \sigma^2_{v_{\text{diff}}^{\text{sh}}}, \mu_{v_{\text{diff}}^{\text{obj}}}, \sigma^2_{v_{\text{diff}}^{\text{obj}}}, \mu_{v_{\text{diff}}^{\text{obj}}}, \sigma^2_{v_{\text{diff}}^{\text{obj}}}, \mu_{v_{\text{diff}}^{\text{obj}}}, \sigma^2_{v_{\text{diff}}^{\text{obj}}}\}$ directly from training data during the offline phase.

OK! Now let's get our hands dirty. First we want to extract the color information and analyze the distributions:

1. Run `make` to compile the code `get_data_under_mask.cc`.
2. Run the Perl script `script-write-hallway-fg-data-to-file.pl` using the command:

   ```
   $ perl script-write-hallway-fg-data-to-file.pl
   ```

   to extract the difference in hue, saturatation, and value components for the objects from the training set.
3. Run the Perl script `script-write-hallway-sh-data-to-file.pl` using the command:

   ```
   $ perl script-write-hallway-sh-data-to-file.pl
   ```

   to extract the difference in hue, saturatation, and value components for the shadows from the training set.
4. To see the distributions over the difference of each component, it would be easy to use Matlab/Octave to plot the data. Therefore, we first need to generate the Matlab/Octave code to plot the distributions from the extracted data. Run

   ```
   $ perl script-gen-data-for-plotting.pl ../data/working/fg_data_file fg
   ```

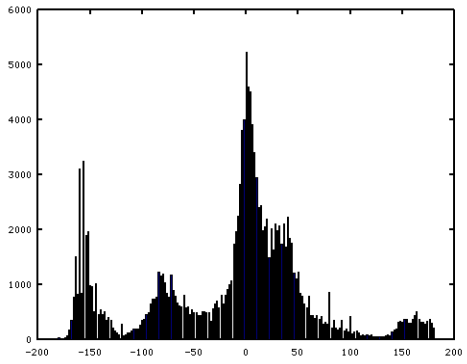

Figure 1: Distribution over the difference in hue component for object pixels.

to generate the plots for object pixels. Also, run

```
$ perl script-gen-data-for-plotting.pl ../data/working/sh_data_file sh
```

to generate the plots for shadow pixels.

5. Run Matlab/Octave under the directory to `../data/working/` then execute the command

   ```
   octave:1> <pixel type>_<component>_file_to_plot
   ```

   Replace `<pixel type>` with `foreground` or `shadow` and replace `<component>` with `h`, `s` or `v`. For example, if you run `foreground_h_file_to_plot`, you should see the distribution similar to Figure 1.

   The above command will also give you the mean and standard deviation of the distribution. You can use them in the next phase.

Online Phase

Given the model estimate Θ, we use the maximum likelihood approach to classify a pixel as a shadow pixel if

$$P(M_{\text{xy}} \mid A_{xy} = \text{sh}; \Theta) > P(M_{\text{xy}} \mid A_{xy} = \text{obj}; \Theta). \tag{3}$$

Otherwise, we classify the pixel as an object pixel. Here we could add the prior probabilities to the shadow model and the object model in (3) to obtain a maximum a posteriori classifier. In this workshop, we can assume equal priors.

To perform the classification:

1. Look at the code `shadow_ml.cc`.
2. Manually set the parameters in the code for the distributions for object pixels and shadow pixels.
3. Run `make` to compile the code.
4. Run the program and see the results.

OK, now we learn how to apply some simple techniques to detect shadows.