A3—Photometric Stereo

CS 43400, Spring 2013

Due on Wednesday March 20 at 7:00am

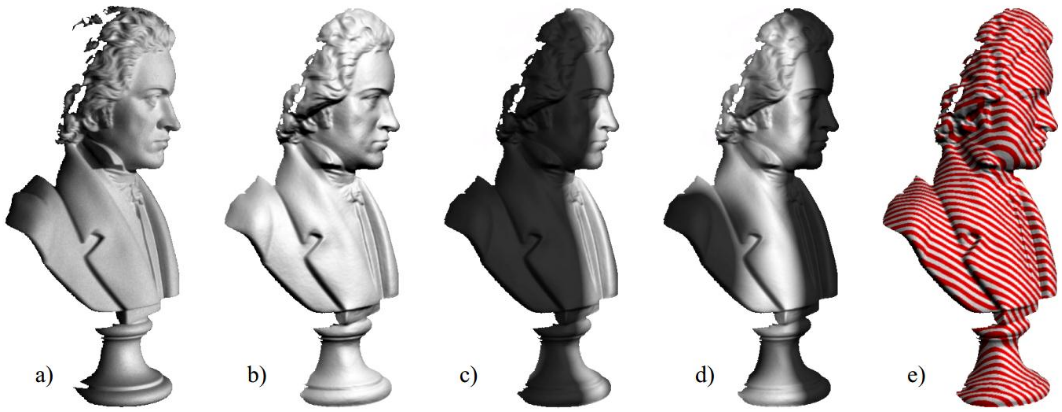
Summary: implement a program that calculates a photometric reconstruction of a simple scene and enables relighting.

1. **General Acquisition Setup**
   a. Using a static camera and a movable light source (e.g., a flashlight, a lamp, etc.) put a close-to-convex diffuse object on a table. Then, place the camera "in front of the object" at a static location. Place the light source in one of 3 or more positions that roughly form a triangle (or square, or pentagon, etc…) in front of the camera and at about the same distance as the camera is to the scene/object. At each light position, take a picture of the scene using the same camera settings (or as close as possible) – do not move the camera! The result will be 3 or more high-resolution pictures of the same object under different illumination conditions. It is to your advantage to choose an interesting, colorful object and to take "good" pictures.
2. **3-image Gray-Scale Photometric Stereo**
   a. Take a set of 3 grayscale images of a scene, using 3 different light directions, and perform a photometric stereo reconstruction. This means each camera pixel will have 3 intensity values. For light directions, use manually estimated vectors (note: remember they should point towards the light and normals should point away from the surface). Estimate the per-pixel normal numerically. You will need to invert a 3x3 matrix. You can use a math library to do such.
   b. Provide an interactive program where you can use the mouse (or a GUI) to alter the illumination direction and then create a newly illuminated scene.

3. **5-image Color Photometric Stereo**
   a. Similar to 3-image case, but take 5 images of the object and ignore the weakest and strongest value for each camera pixel in a simplified effort to omit shadows and any potential specularities. In this case, use the "gray-scale" value to estimate normals but then re-light the intensity of the colored pixels.
   b. Show a similar interactive program but using color.

4. Visualizations

a. Generate at least some simple form of additional re-lighting schemes that yield interesting and compelling visuals. You may make the re-lighting vary depending on the normal direction, local curvature, closeness to end of object (i.e., silhouette) – use your imagination! Below are some tentative results:



a) input image, b) more specular appearance, c) illuminate "far side", d) illuminate "middle range", e) illuminate with a striped red-pattern

5. Extra Credit

a. Expand your method to not require an estimated light direction as input; e.g., enable unknown light direction photometric stereo

b. Develop more complex visualizations to part '4', such as the red-stripes example or others you can imagine

c. Reduce the "noise" in the solution process by ensuring neighboring normals are similar unless explicitly different (e.g., across a discontinuity)

6. Turn in

a. Source code and executable

b. 3 input images for task '2' (labeled testA-1.jpg, testA-2.jpg, and testA-3.jpg)

c. 5 input images for task '3' (labeled testB-1.jpg, testB-2.jpg,…, and testB-5.jpg)

d. Sample screen shots of your visualizations for task '4'.

e. The TA will run your program using testA or testB and will play with interactive re-lighting and your visualizations (make the GUI clear and easy to use).

f. Extra credit as described above, with appropriate extra image files, if any, and instructions.

g. Turn in a single zipped archive with all your files via Blackboard