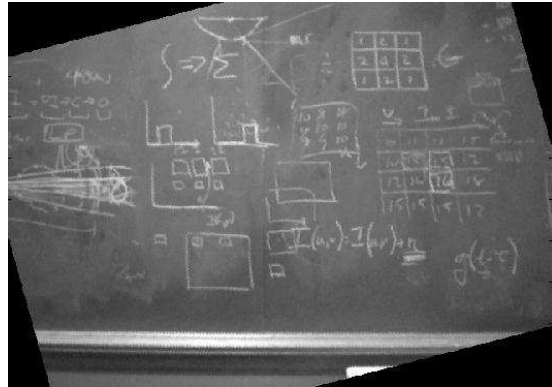
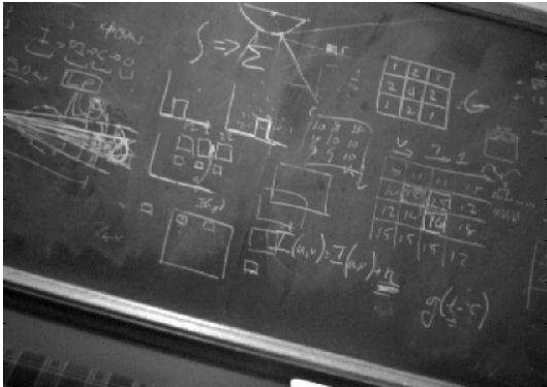


CSE 398/498-010

Real-time Image Processing for Autonomous Robot Systems

Laboratory 8: Image Warping

Report Due Date: 23 Nov 04



A. Objective: Implement image warps for camera translation, rotation, scaling and affine flow models of a planar scene.

B. Lab Procedure:

1. In this lab, you are required to implement two functions:

```
void myAffineWarpBL(MILImage *im1, MILImage *im2, double A[], double T[]);
```

```
void myAffineWarpNN(MILImage *im1, MILImage *im2, double A[], double T[]);
```

These will take source and destination images as inputs, along with the affine transformation matrix A and translation vector T as defined by the relationship

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \left(\begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} t_x \\ t_y \end{bmatrix} \right) = A(\bar{x} - T)$$

Note: Euclidean transformations (rotation, translation, scaling) are merely special cases of affine transformations.

The only difference between the two functions is that the suffixes BL and NN correspond to bilinear and nearest-neighbor interpolation schemes. These techniques are covered in pages 21-23 of the lecture notes.

2. For each of the Euclidean similarity transformations (1. translation, 2. rotation, 3. rotation and translation, 4. rotation, translation and scaling) and a more general affine transformation, you are to perform the respective image warps. **For EACH you will need to save the original image and the warped image FROM THE SAME TIME INSTANT.** Remember that the equations provided assume a planar image scene. Note also that the warps are to be implemented in real-time (NOT on a static image).
3. You should verify the correctness of your warps for the report. For example, set A to be a rotation matrix of $-\theta$, and then rotate the camera until the warped image is “horizontal.” Ensure that the camera rotation is consistent with the warp parameter. **Capture this image.**
4. For the general affine warp, find a matrix A and a camera orientation that will yield a “normal” image after the warp (orthogonal x-y axes that are horizontal and vertical with respect to the image frame). **Identify these and capture the image.**
5. For a given rotation, **record images from bilinear and nearest neighbor interpolation.**

C. Report Requirements:

1. Your code.
2. All images and parameters requested above, along with an accompanying explanation.
3. AT A MINIMUM, answers to the following questions:
 - a. What was the difference in CPU performance between the nearest-neighbor and bilinear interpolation schemes?
 - b. Comment on the effects of the 2 different interpolation strategies on the warped images. It might be useful to zoom in on specific features while doing so.
 - c. Why in all of the warping models was it assumed that the motion of the camera was small?
 - d. Why is it better to compute the reverse transformation when warping the image?
 - e. Your thoughts on improving the process.

f. For the following definitions of the matrix A , what are the effects (mathematically) of applying the below transformations to the image? BE SPECIFIC!

$$\text{i) } \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\text{ii) } \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1/2 & \sqrt{3}/2 \\ -\sqrt{3}/2 & 1/2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\text{iii) } \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\text{iv) } \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\text{v) } \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \sqrt{2}/2 & -\sqrt{2}/2 \\ \sqrt{2}/2 & \sqrt{2}/2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\text{vi) } \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\text{vii) } \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\text{viii) } \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$