

## Solutions to Homework 4

1. A routine that implements the Fourier phase correlation method is:

```
function out = phase(im1,im2)

ft1 = fft2(im1);
ft2 = fft2(im2);
fftratio = (ft1 .* conj(ft2)) ./ abs(ft1 .* conj(ft2));
out = abs(iff2(fftratio));
end
```

We register the coke image to itself using:

```
[i,j] = find(out == max(max(out)))
```

and this yields the result:  $(i,j) = (1,1)$  not  $(0,0)$  as you would expect. This is because matlab indexes things starting from  $(1,1)$ . Other than this, the array does look like a perfect discrete delta function. That is, all the values are zero except for the  $(1,1)$  position which equals 1.

2. Now if we try this with the translated image:

```
out1 = phase(coke,tran); [i,j] = find(out1 == max(max(out1)))
```

we get  $(i,j) = (6,10)$ . Since zero translation comes out as 1, translation by 5 pixels comes out as 6, and translation of 9 pixels comes out 10. This indicates that `tran` is shifted by 5 in y and 9 in x relative to `coke`. This matches our visual expectations.

If we run the phase correlation calculation the other way:

```
out2 = phase(tran,coke);
[i,j] = find(out2 == max(max(out2)))
```

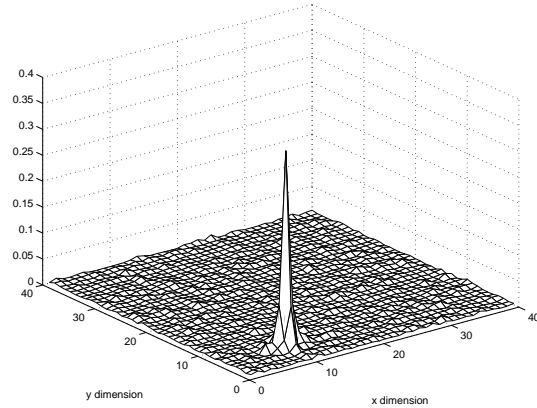
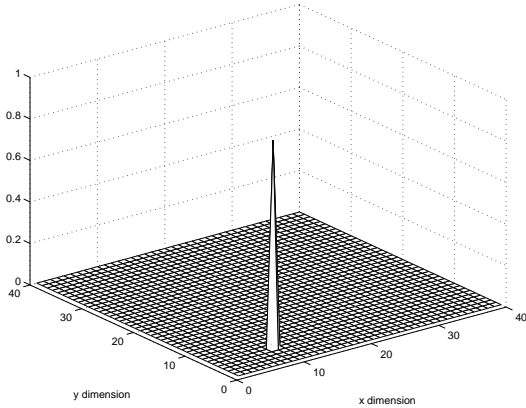
we now get  $(i,j) = (421,219)$ . If `tran` is shifted by  $(5,9)$  relative to `coke`, then you would expect that `coke` is shifted by  $(-5,-9)$  relative to `tran`. But Matlab doesn't have negative indices into arrays. So instead, it's the whole image height -5, and the whole image width -9.

That is, the `tran` image is shifted relative to the `coke` image by 5 pixels upwards  $((425+1) - 421)$ , and by 9 pixels to the left  $((227+1)-219)$ .

The arrays `out1` and `out2` are no longer perfect discrete delta functions, even though the `tran` image is a pure shift of the `coke` image. The output arrays are no longer perfect discrete delta functions because the two images, when they are translated by the correct amount to have perfect alignment, do not match each other exactly. The reason for this is that when you translate an image to the right, what do you put in on the left? As discussed in class, even a pure translation between two images must involve some boundary strips that do not match.

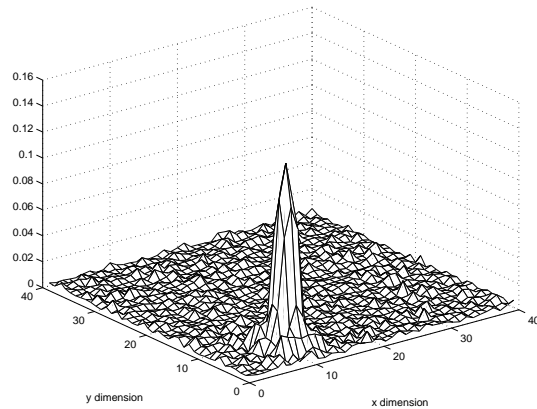
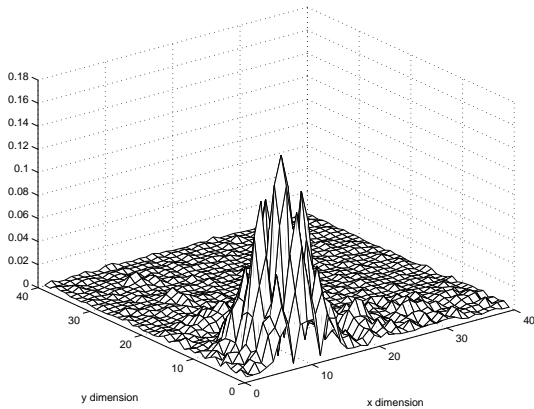
- Registration with the noisy image should give the same results in terms of the actual coordinates of the peak at (6,10). The registration algorithm is robust against this type and amount of noise.

Here are some pictures showing what the spike looked like for the cases coke/tran and coke/noisy. The plots were generated by the command `mesh ( out ( 1 : 40 , 1 : 40 ) )`. I only plotted the relevant area of the output to save printing time.



- Again, the registration produces the same result.

- The version that was both noisy and blurred still should yield the same translation parameters. But perhaps for some people, the noisy/blurred one will be one pixel or so off. Here are the meshplots for the registrations of coke/blurred and coke/realbad.



- For this problem, you were asked to replace parts of the image, or in some other way, do something drastic to the image and see if it thwarts the phase correlation method. As it turns out, this method is very robust.
- Last problem: Certainly for pure translation, both methods work. When things get noisy/blurred/messy, the regular method is better, possibly because when you just do  $F_2/F_1$  you can have instability problems (it blows up when  $F_1$  becomes close to zero) whereas the regular method doesn't have this same problem, since the numerator and the denominator have the same magnitude.