Super-resolution motion deblurring

Physics 305 Activity 1
2nd Sem AY 2022-2023

Mark Jeremy G. Narag
2014-64423
PhD Physics

All the codes and files in this activity are available on my Github:

https://github.com/mgnarag/physics305_computational_imaging

This is a screenshot of a **still** toy car in a video…

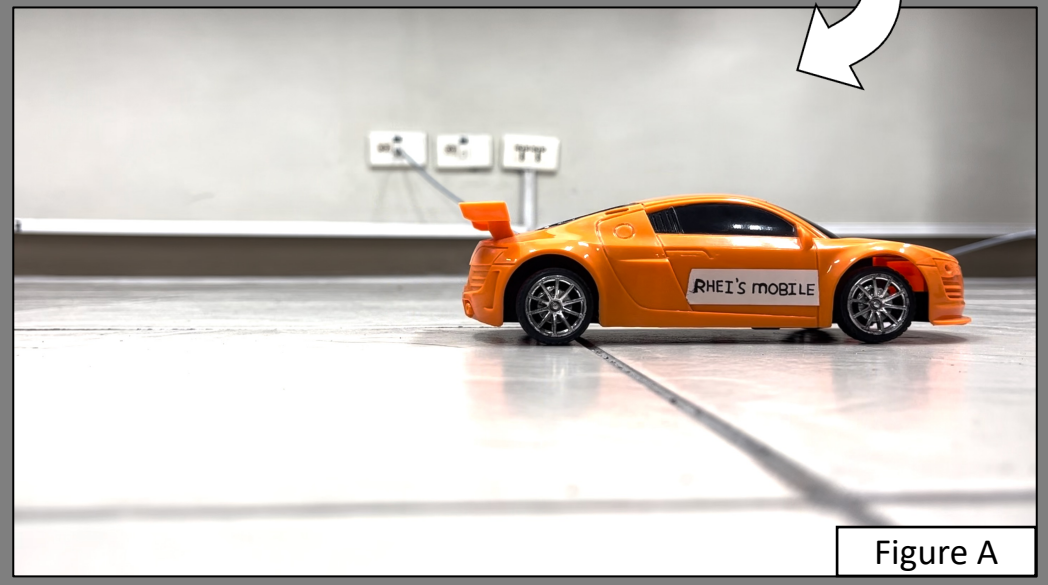


Figure A

…and this is a screenshot of the **moving** toy car in a video… it's blurred!

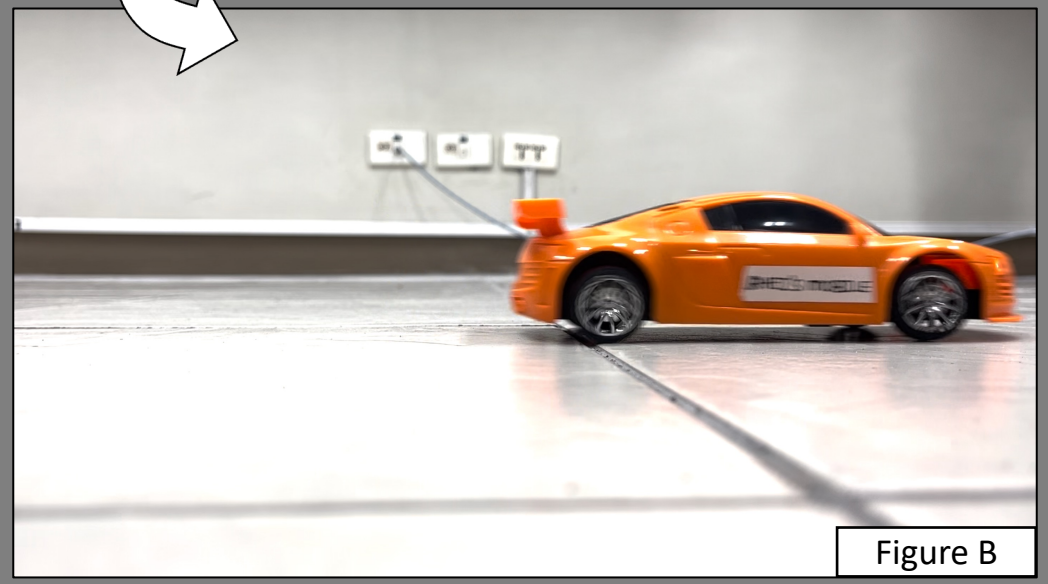


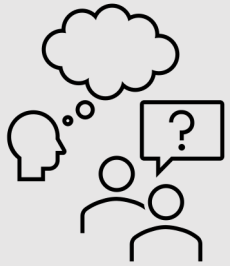
Figure B

How can we deblur an image?

Imagine we have an object or a scene [Figure A, let it be *f(x,y)*]. This object then moves, and our camera phone captures it. Of course, we won't see a clear image but rather a distorted or blurred version of the object [Figure B, let it be *g(x,y)*].

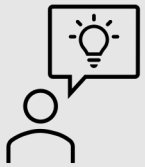
This output image $g(x,y)$ can be represented as the convolution of the point spread function of the camera $h(x,y)$ and the object $f(x,y)$ plus noise $n(x,y)$ [1]:

$$g(x,y) = h(x,y) * f(x,y) + n(x,y)$$



So for example we have a blurred image $g(x,y)$ just like Figure B, how can we deblur it? Or in other words, how can we reconstruct $f(x,y)$?

One way is through….



Minimum Mean Square Error (Wiener) filtering

[1] Soriano, M. 2023. Super-resolution Motion Deblurring. Physics 305 Computational Imaging

This is a screenshot of a **still** toy car in a video...

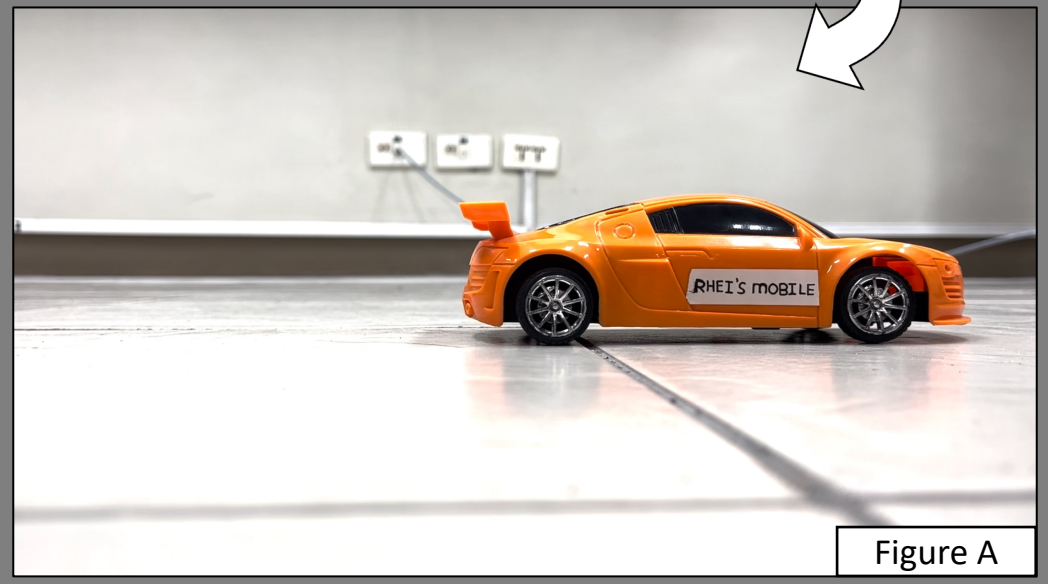


Figure A

...and this is a screenshot of the **moving** toy car in a video... it's blurred!

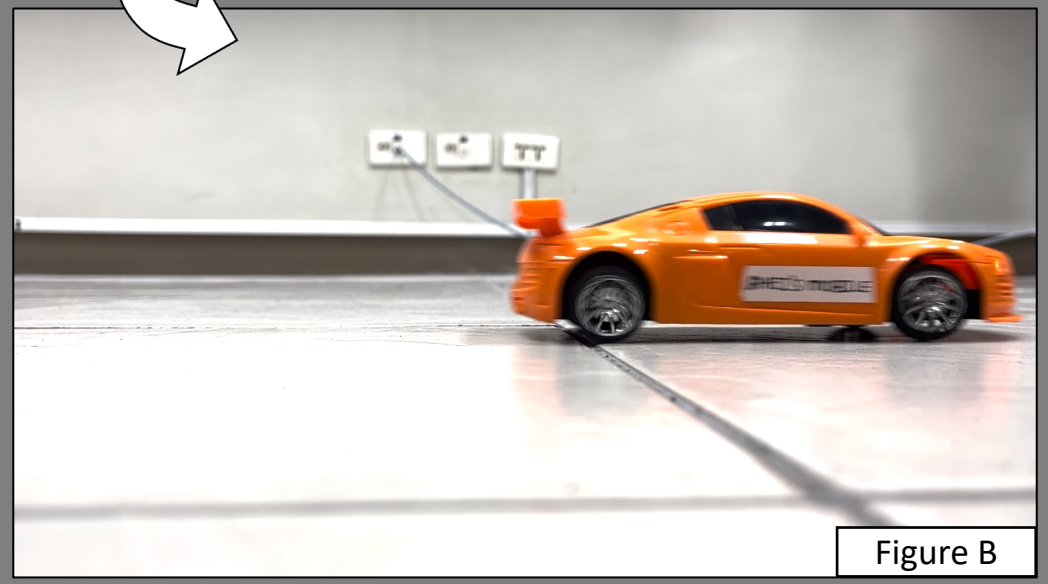


Figure B

How can we deblur an image?

In Minimum Mean Square Error (Wiener) filtering, we want to minimize the square error function:

$$e^2 = E\{(f - \hat{f})^2\}$$

E{} is the expectation, $\hat{f}$ is an estimate, f is the object/scene. In frequency domain, the solution that minimizes this function is:

$$\hat{F}(u,v) = \left[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2 + S_n(u,v)/S_f(u,v)}\right] G(u,v)$$

$\hat{F}(u,v) \rightarrow Fourier\ Transform\ of\ \hat{f}(x,y)$
$H(u,v) \rightarrow Fourier\ Transform\ of\ h(x,y)$
$G(u,v) \rightarrow Fourier\ Transform\ of\ g(x,y)$
$S_n(u,v) \rightarrow power\ spectrum\ of\ the\ noise$
$S_f(u,v) \rightarrow power\ spectrum\ of\ the\ object$

Let's turn the ratio of $S_n(u,v)/S_f(u,v)$ as single number called NSR (noise-to-signal ratio) [1].

Now, let's try to do some deblurring!

[1] Soriano, M. 2023. Super-resolution Motion Deblurring. Physics 305 Computational Imaging

Deblur images using Wiener filter

There is already a Matlab functions that performs Wiener filtering – deconvwnr:

J = deconvwnr(I,psf,nsr)

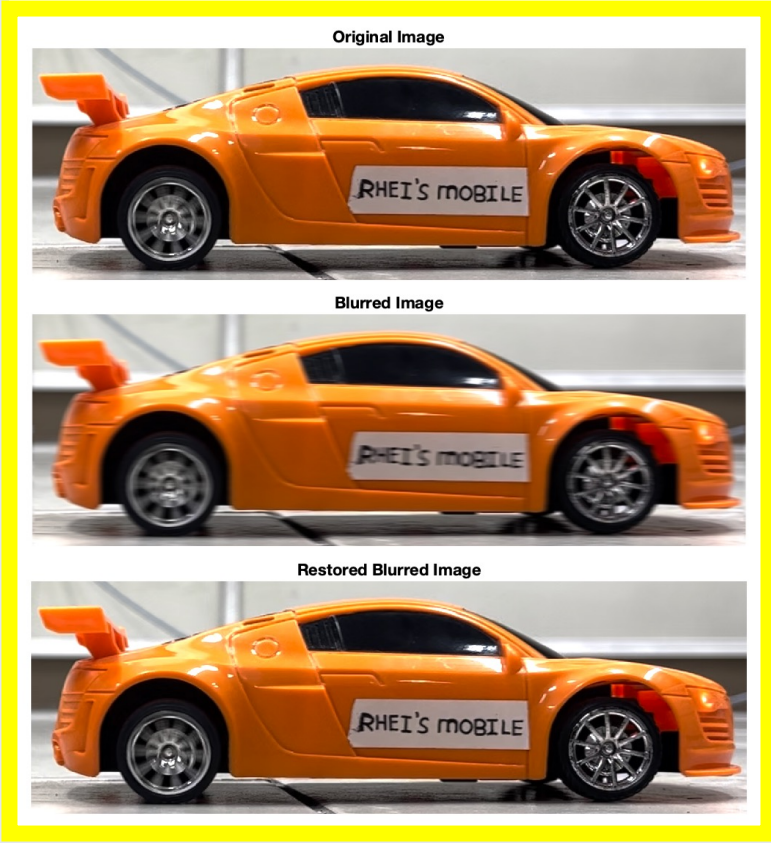
The above function deconvwnr returns a deblurred image J of a blurred image I. Image I is being convolved with the point-spread function psf. In addition, there is an additive noise given by the noise-to-signal power ratio nsr [2].

Here is an example of a Matlab code that uses the above function. Now, let's apply it to our own pictures. Before we use blurred images tho, let's try synthetically blur a clear image first. We should be able to reconstructed the original.

[2] https://www.mathworks.com/help/images/ref/deconvwnr.html

MATLAB code

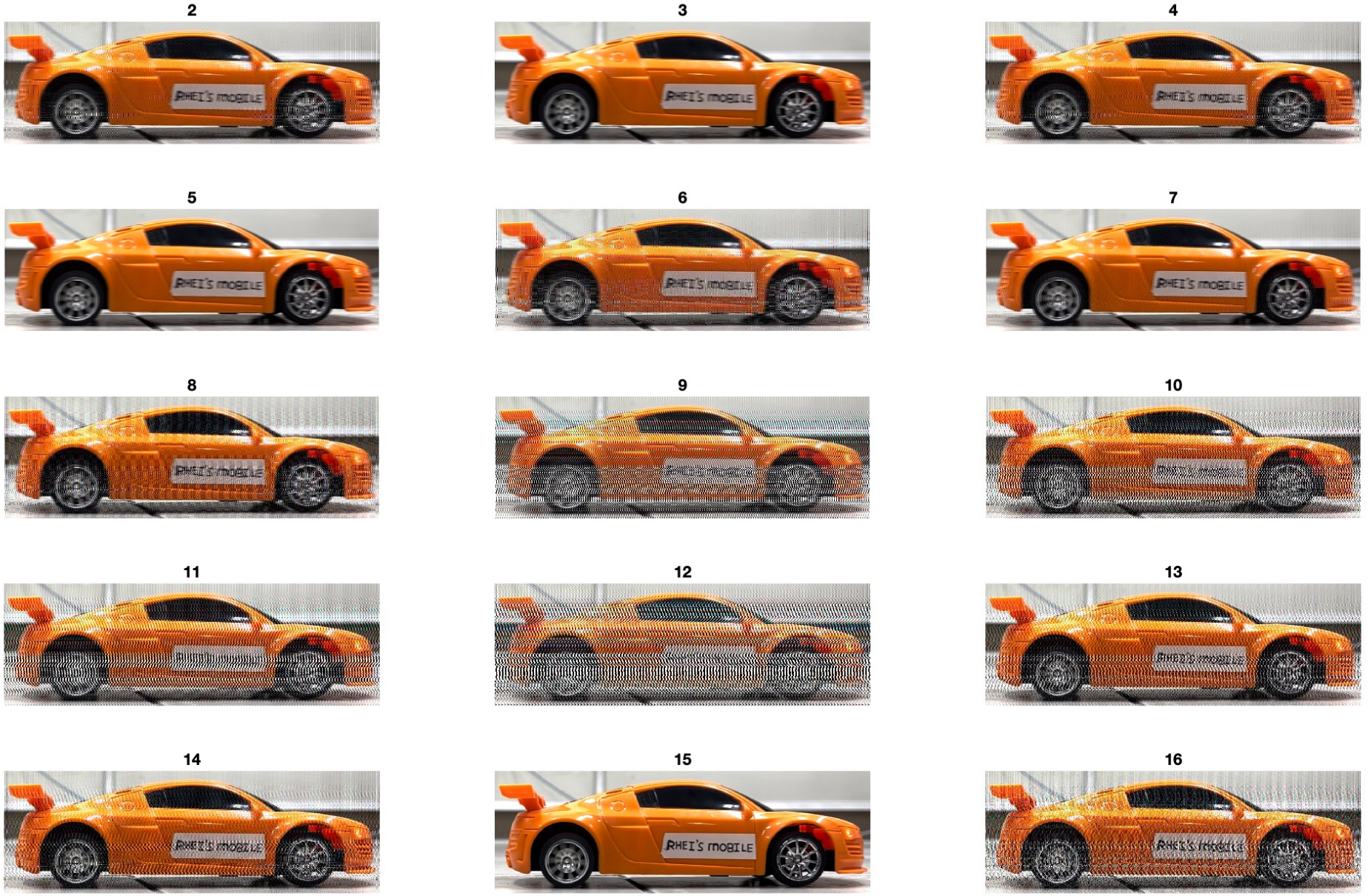
```
figure();
Ioriginal = imread('non_blur.png');
subplot(131); imshow(Ioriginal); title('Original Image')

PSF = fspecial('motion',15,0);
Idouble = im2double(Ioriginal);
blurred = imfilter(Idouble,PSF,'conv','circular');
subplot(132); imshow(blurred); title('Blurred Image')

wnr1 = deconvwnr(blurred,PSF);
subplot(133); imshow(wnr1); title('Restored Blurred Image')
```

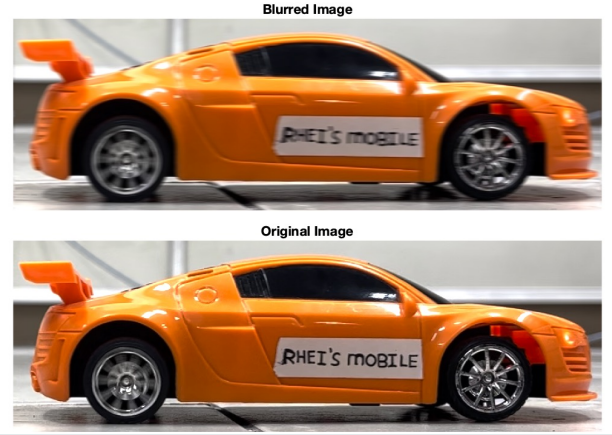


Shown in Figure 1 is an example of *synthetically* blurring Rhei's mobile car then deblurring it. We did not add noise here yet. As we can see, we were able to successfully deblur it because we already know the PSF that we used for blurring it.

Figure 1. Blurring and deblurring a still image. Blurring the original image with PSF pixel offset = 15, angle = 0

Part 1: Deblurring a synthetic blurred image





But how about when we don't know the PSF? And we are left to randomly guess the pixel offset and angle? Shown in Figure 2 is the restored blurred images for varying pixel offset. As we can see, if we don't know the PSF, it will be a trial-and-error approach. One can pre-conclude that the best reconstruction is on PSF pixel offset = 7.

Figure 2. "Guessing" the correct pixel offset by manually investigating the restored version from different values of pixel offset

Part 2: Deblurring a real blurred image

Now let's try the blurred a while ago (Figure B):

This is not a synthetically blurred image. This is real. It is a screenshot from a video. We really don't know the PSF on this one, so we are left with randomly guessing the best combination.

On the right is the MATLAB code to do it.

MATLAB code

```
Blurred_img = imread('Blur_test_2.png');
figure(); t = tiledlayout(10,6,
'TileSpacing','none','Padding','compact');



for pix=23:1:32
    for NSR = 0:0.01:0.05
        PSF = fspecial('motion',pix,0);
        wnr1 = deconvwnr(Blurred_img,PSF,NSR);
        nexttile; imshow(wnr1);
        title(['PSF=',num2str(pix),',NSR=',num2str(NSR)]
        )
    end
end
```

Part 2: Deblurring a real blurred image

As we can see in Figure 3, it seems that we can't deblur the image using only the pixel offset as our parameter. Let's try varying the NSR as well.

We need not to explore the PSF angle since the phone was kept on the ground still during recording.

Figure 3. Varying PSF pixel offset for NSR = 0, PSF type = 'motion', PSF angle = 0.

Part 2: Deblurring a real blurred image

Here are the results for varying PSF and NSR (more on the next slides)

Reference:

Figure 4. Varying PSF pixel offset and NSR

Reference: 

NSR

PSF

PSF=12,NSR=0	PSF=12,NSR=0.01	PSF=12,NSR=0.02	PSF=12,NSR=0.03	PSF=12,NSR=0.04	PSF=12,NSR=0.05
PSF=13,NSR=0	PSF=13,NSR=0.01	PSF=13,NSR=0.02	PSF=13,NSR=0.03	PSF=13,NSR=0.04	PSF=13,NSR=0.05
PSF=14,NSR=0	PSF=14,NSR=0.01	PSF=14,NSR=0.02	PSF=14,NSR=0.03	PSF=14,NSR=0.04	PSF=14,NSR=0.05
PSF=15,NSR=0	PSF=15,NSR=0.01	PSF=15,NSR=0.02	PSF=15,NSR=0.03	PSF=15,NSR=0.04	PSF=15,NSR=0.05
PSF=16,NSR=0	PSF=16,NSR=0.01	PSF=16,NSR=0.02	PSF=16,NSR=0.03	PSF=16,NSR=0.04	PSF=16,NSR=0.05
PSF=17,NSR=0	PSF=17,NSR=0.01	PSF=17,NSR=0.02	PSF=17,NSR=0.03	PSF=17,NSR=0.04	PSF=17,NSR=0.05
PSF=18,NSR=0	PSF=18,NSR=0.01	PSF=18,NSR=0.02	PSF=18,NSR=0.03	PSF=18,NSR=0.04	PSF=18,NSR=0.05
PSF=19,NSR=0	PSF=19,NSR=0.01	PSF=19,NSR=0.02	PSF=19,NSR=0.03	PSF=19,NSR=0.04	PSF=19,NSR=0.05
PSF=20,NSR=0	PSF=20,NSR=0.01	PSF=20,NSR=0.02	PSF=20,NSR=0.03	PSF=20,NSR=0.04	PSF=20,NSR=0.05
PSF=21,NSR=0	PSF=21,NSR=0.01	PSF=21,NSR=0.02	PSF=21,NSR=0.03	PSF=21,NSR=0.04	PSF=21,NSR=0.05

Reference: 

Highlighted in yellow box is the best one yet.

NSR

PSF

PSF=23,NSR=0	PSF=23,NSR=0.01	PSF=23,NSR=0.02	PSF=23,NSR=0.03	PSF=23,NSR=0.04	PSF=23,NSR=0.05
PSF=24,NSR=0	PSF=24,NSR=0.01	PSF=24,NSR=0.02	PSF=24,NSR=0.03	PSF=24,NSR=0.04	PSF=24,NSR=0.05
PSF=25,NSR=0	PSF=25,NSR=0.01	PSF=25,NSR=0.02	PSF=25,NSR=0.03	PSF=25,NSR=0.04	PSF=25,NSR=0.05
PSF=26,NSR=0	PSF=26,NSR=0.01	PSF=26,NSR=0.02	PSF=26,NSR=0.03	PSF=26,NSR=0.04	PSF=26,NSR=0.05
PSF=27,NSR=0	PSF=27,NSR=0.01	PSF=27,NSR=0.02	PSF=27,NSR=0.03	PSF=27,NSR=0.04	PSF=27,NSR=0.05
PSF=28,NSR=0	PSF=28,NSR=0.01	PSF=28,NSR=0.02	PSF=28,NSR=0.03	PSF=28,NSR=0.04	PSF=28,NSR=0.05
PSF=29,NSR=0	PSF=29,NSR=0.01	PSF=29,NSR=0.02	PSF=29,NSR=0.03	PSF=29,NSR=0.04	PSF=29,NSR=0.05
PSF=30,NSR=0	PSF=30,NSR=0.01	PSF=30,NSR=0.02	PSF=30,NSR=0.03	PSF=30,NSR=0.04	PSF=30,NSR=0.05
PSF=31,NSR=0	PSF=31,NSR=0.01	PSF=31,NSR=0.02	PSF=31,NSR=0.03	PSF=31,NSR=0.04	PSF=31,NSR=0.05
PSF=32,NSR=0	PSF=32,NSR=0.01	PSF=32,NSR=0.02	PSF=32,NSR=0.03	PSF=32,NSR=0.04	PSF=32,NSR=0.05

Blurred image:

Figure 5 shows the best reconstructed deblurred image that we found so far. Comparing to the original blurred image, although not perfect with some artifacts present, **we were now able to read the letters!**

In conclusion, we can say that it is really hard to reconstruct a blurred image rather than a synthetically blurred image. It is hard to guess the right parameters and the reconstruction, by far, is judged visually. There is a very large range of values of the parameters (PSF and NSR).

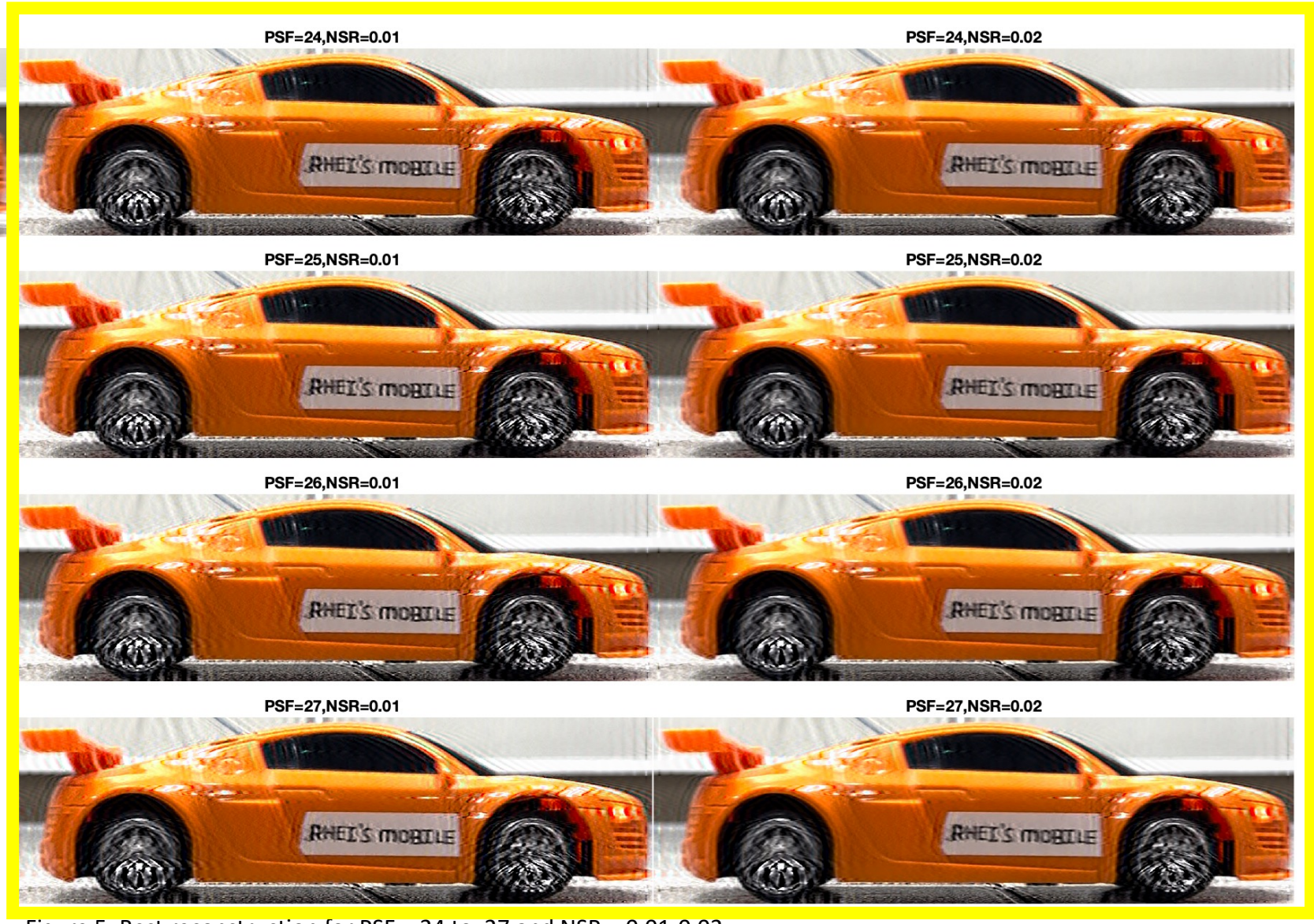


Figure 5. Best reconstruction for PSF = 24 to 27 and NSR = 0.01-0.02

Debluring through stacking and aligning

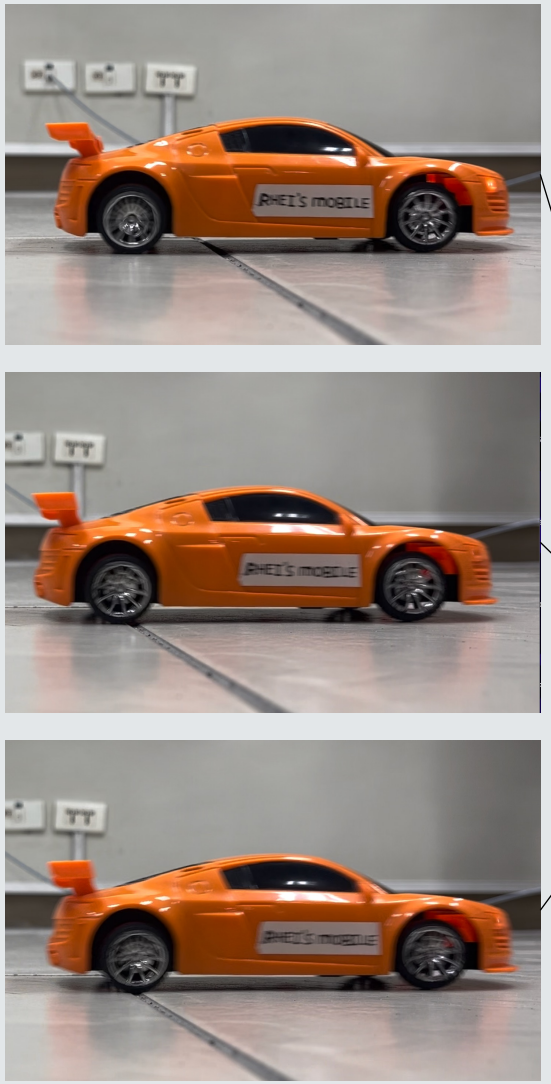


Figure 6. 3 sample images used in stacking

Another way of deblurring and image is by stacking them. Say for example you have N low resolution images of the same scene. By stacking them, we can super-resolve the scene. We can do this in GIMP where we can set the opacity of each image to low then align the scenes for all the frames. Shown below is an example of the super-resolved Rhei's automobile:

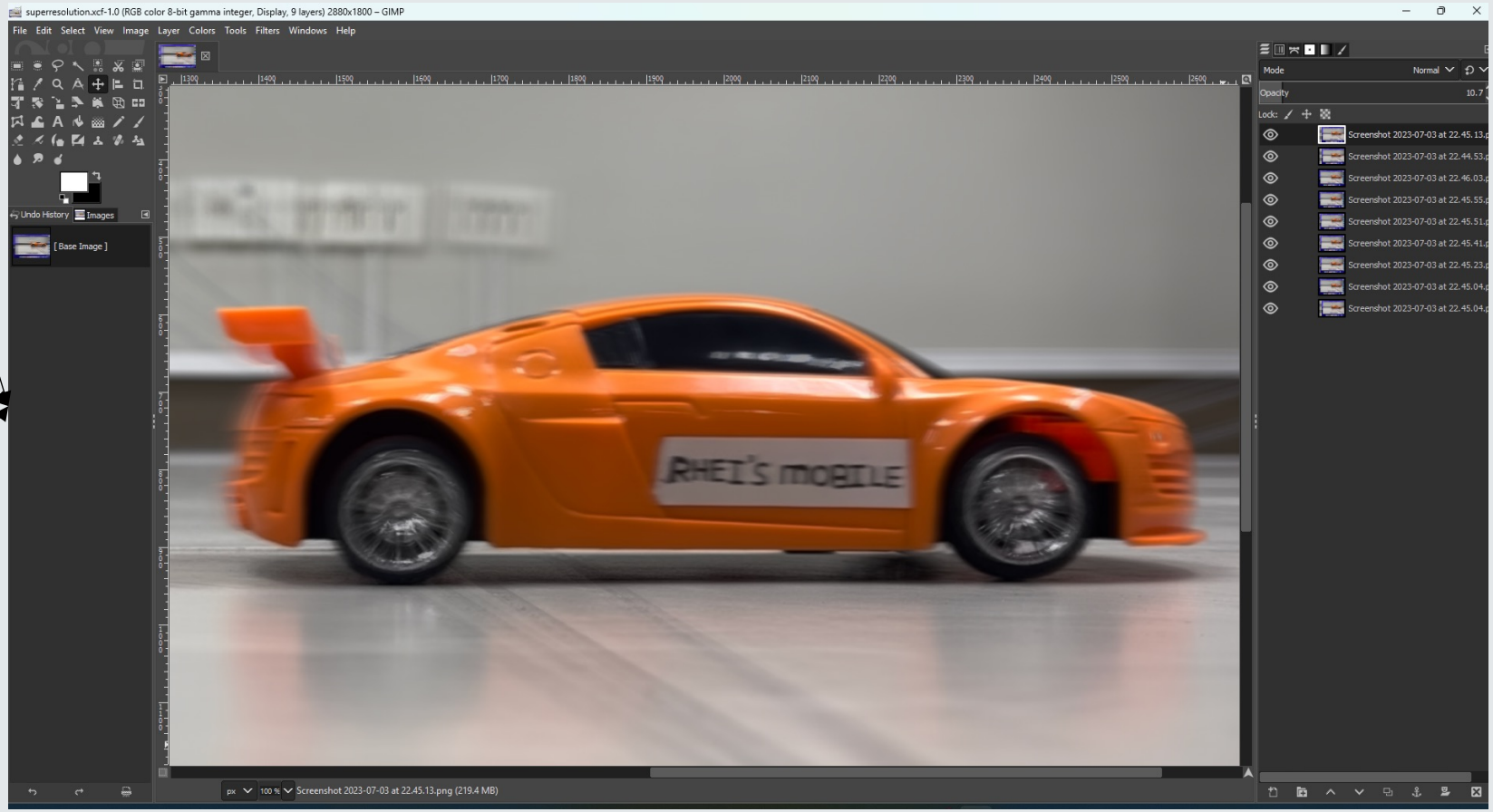


Figure 7. Stacked and aligned nine (9) images of the same scene.

Debluring through stacking and aligning

We can also do this on images with multiple shots:

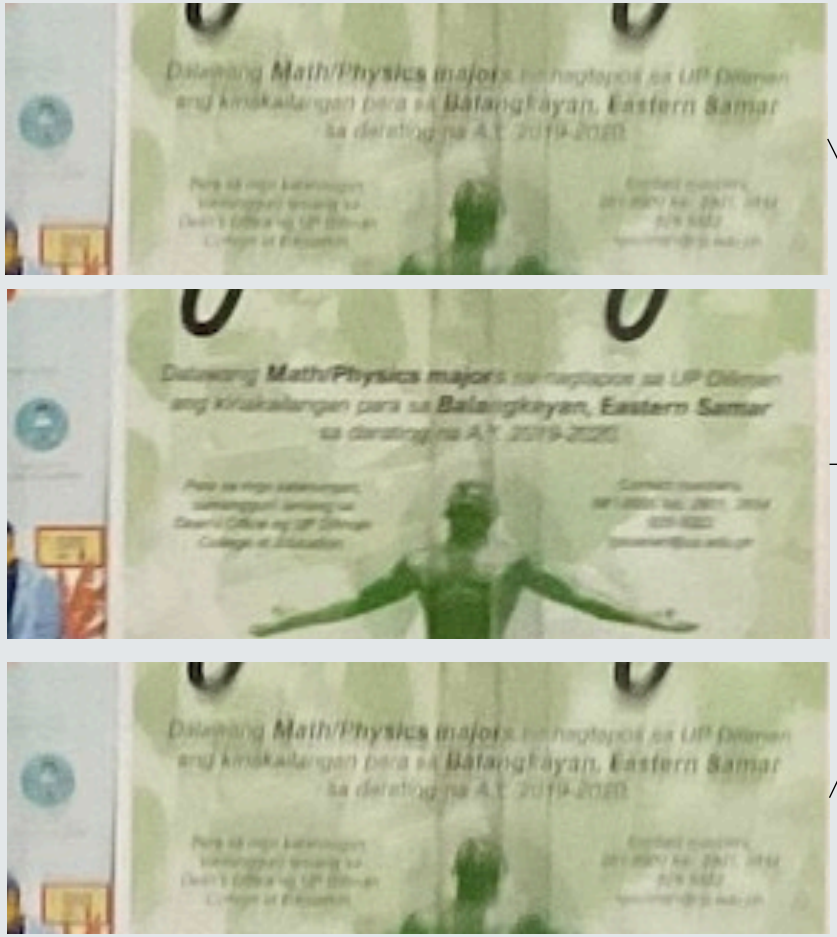


Figure 6. 3 sample images used in stacking

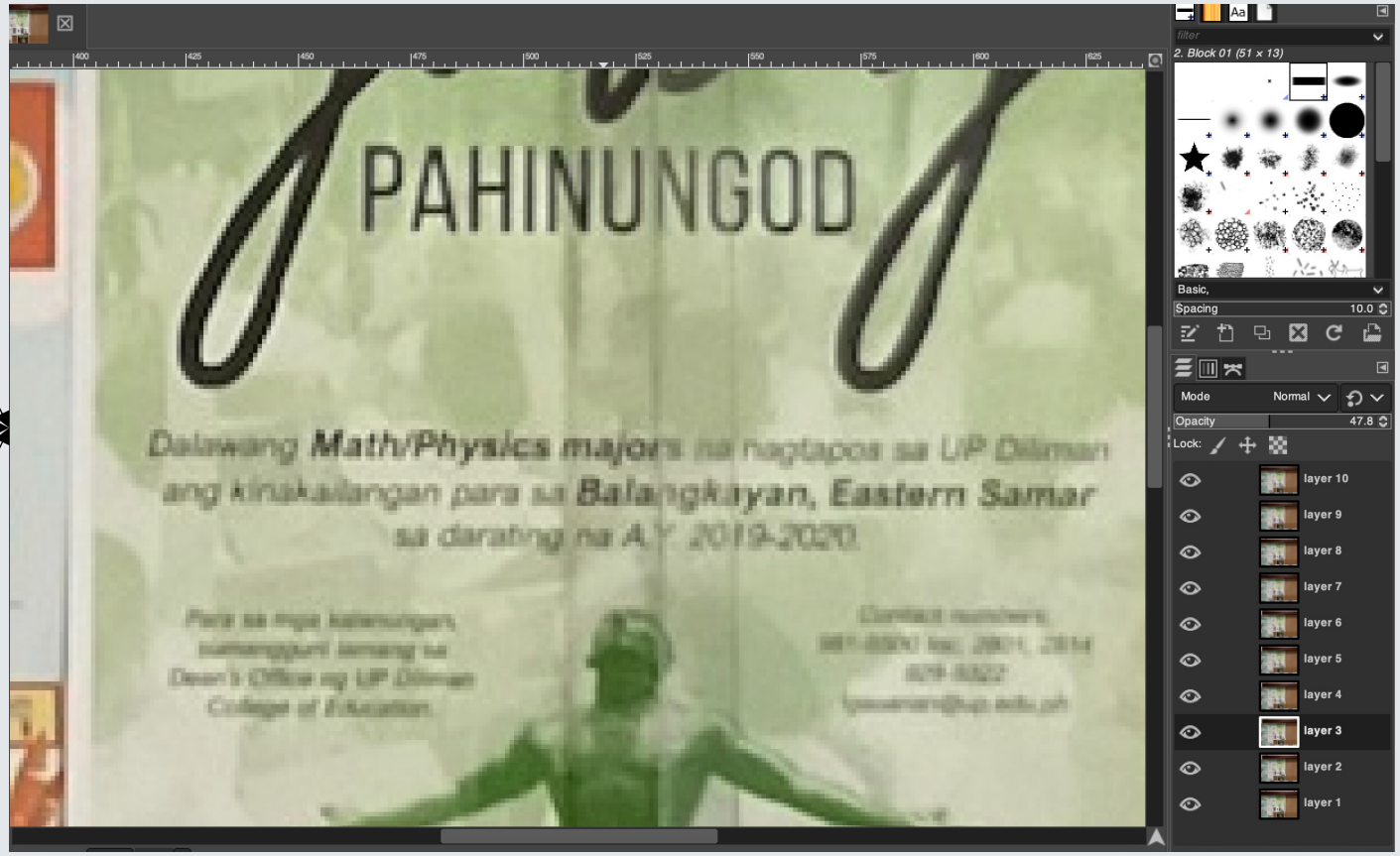


Figure 7. Stacked and aligned ten (10) images of the same scene.

THE WORDS ARE NOW READABLE!

Reflection

I enjoyed doing this activity especially when I was stacking and realigning the photos then you can already see that the image becomes sharper!

I actually did this way back in March when it was given and was *happy* because during that time, some of my classmates were not able to reconstruct it well using Wiener (to clear it up, I'm not happy because they failed, I am happy because mine worked). Turns out its their captured video itself. My phone have its own stand that's why I was able to make it stable during recording. I then shared my video to my classmates (wholeheartedly).

CRITERIA	QUALIFICATIONS	SCORE
Technical correctness	• Met all objectives. • Theory is discussed sufficiently. • Procedures and Results are complete. • Procedures and Results are verifiably correct. • Understood the lesson.	40
Quality of presentation	• All text and images are of good quality. • Code has sufficient comments and guides. • All plots are properly labeled and are visually understandable. • The report is clear.	30
Self-Reflection	• Explained the validity of results. • Discussed what went right or wrong in the activity. • Justified the self-score. • Acknowledged sources (e.g. persons consulted, references, etc.)	30
Initiative	• Experimented beyond what was required. **Somehow** • Made significant improvements to existing code. **No** • Analyzed limitations or potential of technique, etc. **Yes**	4

94