# Reconstruct a Hyperspectral Image from a Single Shot

Yixuan Shao

**Abstract**—Conventional hyperspectral imaging methods usually suffer from problems such as low temporal and spatial resolution, requiring complicated and expensive optics, and low signal-to-noise ratio. In this project, by repeating the work in [1], I realize a hyperspectral image reconstruction algorithm with high spatial and spectral resolution from a single shot. By placing a dispersive optic in front of an conventional 3-channel camera, the PSF would be horizontally dispersive and there would be dispersion at edges. With the prior knowledge of small total variation and spectral smoothness, reconstructing the hyperspectral image from dispersion at the edges is formed as a 3-step inverse problem. Solving the inverse problem using the conjugate gradient method and alternating direction method of multipliers, I reconstruct the spectra from 430 to 650 nm with 10 nm intervals. The results analysis show that the algorithm produces hyperspectral images with both high PSNR and low spectral error, and is robust under a high noise level.

**Index Terms**—Hyperspectral Imaging, Inverse Problem, Computational Photography

## 1 Introduction

HYPERSPECTRAL analysis is useful in many applications such as material classification, remote sensing, biomedical diagnosis, and image segmentation [2], [3], [4]. However, most modern cameras only have three color channels and can only provide very coarse information about the full spectrum. For example, two materials can display the same color but consist of quite different spectra [5].

Existing hyperspectral imaging methods, such as using a set of band-pass filters [6], [7], a mask and collimating optics [8], are usually bulky, expensive, and suffer from low spatial resolution or temporal resolution. So these methods are not affordable or practical for ordinary customers.

To avoid these problems, in this project I repeat a novel single-shot hyperspectral imaging technique proposed in [1]. This technique only needs to place a prism in front of a conventional 3-channel DSLR camera to generate dispersion, and use a compressed-sensing-based algorithm to reconstruct the hyperspectral image. This algorithm doesn't sacrifice spatial or temporal resolution for spectral information. All it needs is a prism and conventional 3-channel DSLR camera, which are cheap. We could reconstruct the full-size hyperspectral image from a single shot and produce a corresponding RGB image with the full resolution of the imaging sensor.

The major procedures of the hyperspectral image reconstruction algorithm are summarized in Fig. 1. As we place a dispersive prism in front of the camera, the captured image is a dispersed RGB image, as shown in (b). We can see some dispersion at the edges of the two lemons, which contains cues of the spectra. So the remaining steps are inferring the spectra of the whole image from the edges. I first align the captured image based on the image formation model, and we get (c). Then I apply an edge detector to extract the edge pixels which contain more spectral information than interiors, as (d) shows. Afterward, I extract the spectra at edges in the gradient domain, and finally reconstruct the hyperspectral image from it. (g) shows the spectra of the fake lemon and the real lemon. We can see there are more long-wave components in the reconstructed spectrum of the real lemon, which accords with the ground truth spectrum. Therefore, we can discriminate different materials based on this hyperspectral image reconstruction algorithm.

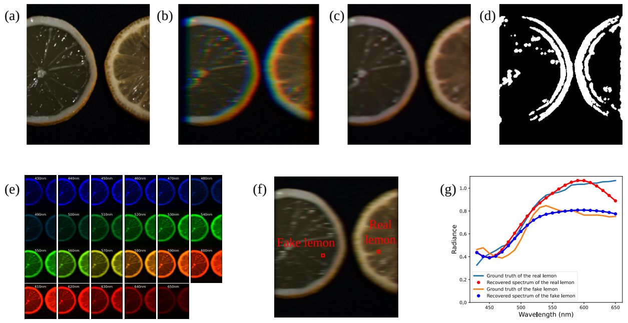


Fig. 1. By adding dispersion to the image formation process, we could reconstruct the hyperspectral images with both high spatial resolution and high spectral accuracy. (a) RGB version of the raw image. (b) Captured dispersed RGB image. (c) Aligned image in RGB color space. (d) Detected edge based on the aligned RGB image. (e) Reconstructed hyperspectral image in each wavelength channel. (f) Recovered image in RGB color space. (g) Comparison of the reconstructed spectra and ground truth spectra of the fake lemon and the real lemon in red rectangular regions in (f).

## 2 Related Work

*Filter-based hyperspectral imaging.* A straightforward way of obtaining hyperspectral images would be using a set of band-pass filters [6], [7], [9]. This comes in two different ways. One way is to place a band-pass filter in front of the camera and switch to a different filter for each shot. So in each shot the camera captures the image of a specific narrow wavelength band. Then the hyperspectral image is formed by combining tens of images. However, this method requires more than one shot and thus the temporal resolution can be low when recording a video is needed. In addition, the mechanical filter switching motion can cause blur.

Another way is to replace the Bayer color filter array with a complicated filter array composed of filters of more bands. However, the effective pixel number for each wavelength band decreases then. The more bands we need, the lower spatial resolution would be, and the more mosaic the image would be. Besides, filters with a narrow band can have very low transmittance, which can decrease the signal-to-noise ratio.

*Mask-based hyperspectral imaging*. A typical mask-based hyperspectral imaging system is implemented in [8]. The idea is to use a prism to generate spatial dispersion. To avoid overlapping in the spectra of neighboring points and increase spectral accuracy, the researchers place an occlusion mask to sample the scene sparsely. This method has high spectral precision. However, the mask and collimating optics require very careful calibration and are expensive. Besides, the sparsely sampled scene suffers from a low spatial resolution.

The method I use in this project is similar in extracting spectrum from dispersion. However, as I make use of the prior knowledge of the small total variation and spectra smoothness to solve the hyperspectral image from an inverse problem, the mask and collimating optics are not needed. Besides, the hyperspectral image produced by my method has a very high resolution.

## 3 Proposed Method

In this section, I will first explain the dispersive image formation model of this hyperspectral camera. Then I will describe the implemented algorithm that reconstructs the hyperspectral image from the dispersed RGB image.

### 3.1 Dispersive image formation model

In [1], the researchers place a prism in front of the camera to generate dispersion. The prism causes images of different wavelengths to shift along a certain direction. The shift amount is determined by the wavelength and the dispersion property of the prism. To avoid the tedious camera calibration process and simplify the procedure but without loss of generality, I arbitrarily assign a dispersive image formation model for this project. The forward model is summarized in Fig. 2 [1].

#### 3.1.1 Model formation

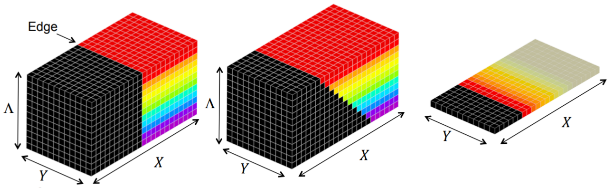
The first step of the forward model is to shift the image based on wavelength, which mimics the dispersion caused by the prism. The raw hyperspectral images from the dataset contain images whose wavelength ranges from 400 to 700 nm with a spacing of 10 nm. I shift the image along the x-axis by 1 pixel every 10 nm wavelength. So the number of wavelengths $\Lambda$ is 31. But for spectral accuracy concern, the reconstructed spectra will only cover 430 to 650 nm.

The second step is to project the hyperspectral image to an RGB image, which mimics the 3-channel sensing function of a conventional RGB camera. The dispersed hyperspectral image is first projected to XYZ color space based on the standard CIE 1931 color matching system. Then I convert the image from XYZ color space to RGB color space.

Both the two steps above are linear transformations. So I can use a matrix multiplier $\mathbf{F}$ to denote the two steps above, which is short for forward model. Finally, I add some Gaussian noise $\eta$ to the captured RGB image. So the forward model could be represented as

$$\mathbf{j} = \mathbf{F}\mathbf{i} + \eta, \tag{1}$$

where $\mathbf{j} \in \mathbb{R}^{XY3\times 1}$ and $\mathbf{i} \in \mathbb{R}^{XY\Lambda\times 1}$ are the vectorized captured RGB image and the raw hyperspectral image, respectively.



(a) Original spectral cube (b) Dispersed spectrum (c) Projected dispersion

Fig. 2. Forward model. Adapted from [1]. (a) The raw hyperspectral image input is non-dispersive. (b) By dispersion, images will shift spatially based on the wavelengths. (c) The dispersed hyperspectral image is captured by a conventional 3-channel camera, which is equivalent to converted into the RGB color space.

#### 3.1.2 Insight into the model

The forward model is basically a lossy compression of the raw image. The raw hyperspectral image contains 31 wavelength channels, while the captured images only contain 3 color channels. Although a lot of information is lost in the forward model and it seems hard to reconstruct the spectra under such a high information compression ratio, the key insight is that the raw image is usually piece-wise smooth. So the spectral information from the interior of a smooth piece can be inferred from the edge. Thus we could compress information from interiors more and save more information for the edge. By adding a prism in front of the lens system, the spectrum at the edge could penetrate into the interior of a smooth piece. That to say, we sacrifice the pixels of the interior of a smooth piece for more spectral information of edges. This is similar to the idea behind many compression algorithm, such as Huffman coding, where data with high-frequency occurrence is stored with fewer bits [10]. In this image formation model, the interiors of the images, which is usually homogeneous in their spectra, are mapped into fewer pixels. The edges, which usually consists of varying spectra, are mapped into more pixels. Therefore it's still highly possible to reconstruct the spectra under such a high information compression ratio.

Fig. 3 illustrates the intuition of why dispersion could provide spectral cues and how to design the reconstruction algorithm [1]. The left top image is a non-dispersed image. Because of metamerism, we cannot restore the spectrum from a non-dispersed image. The left image in (b) shows the spatial gradient of the same image along the x-axis. The figure below is the plot of the gradient of RGB channels in the orange line. We can see the gradient is non-zero only at the edge. But because the gradients in different channels are aligned, it's hard to extract the full spectrum based on a single non-zero pixel value. On the other hand, the right column shows the case of the corresponding dispersed image. We can see the gradient values of the interiors are still zero. That's because the spectra of neighboring pixels are the same, and when they overlap, they together recover the same original color and produce zero gradients. The zero gradients cannot provide spectral cues. However, we can see a smooth change of intensity at the edge. The gradient values are strongly related to the spectra. That indicates there's abundant spectral information at the edges. So this intuition gives us a hint on how to reconstruct the raw hyperspectral image. We could first recover the spectra of edge pixels in the gradient domain. Then we could infer the spectra of interiors based on spectra at edges.

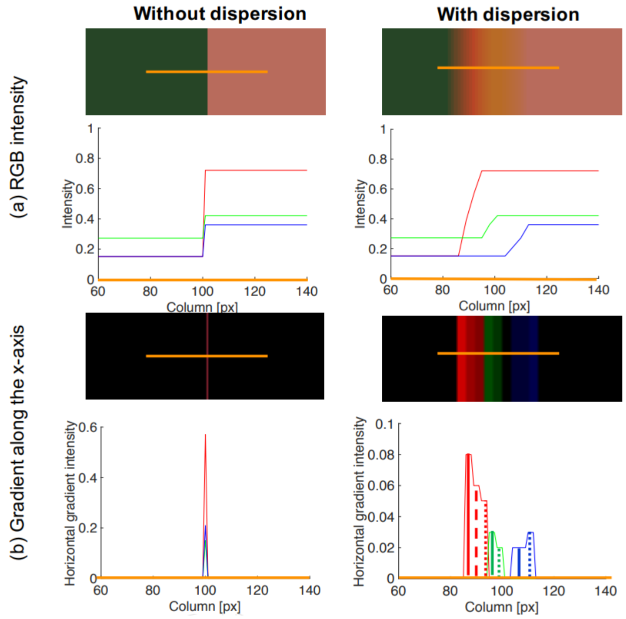


Fig. 3. Adapted from [1]. (a) Left: Non-dispersed image; right: dispersed image. (b) The corresponding x-axis gradient image of the above images. Orange lines refer to the plots of pixel intensities below.

### 3.2 Hyperspectral image reconstruction algorithm

The hyperspectral image reconstruction algorithm consists of three major steps. As mentioned before, there are more spectral cues at the edge. So the first step is to restore the aligned image, which is not dispersed, so that we can then detect the edges. Secondly, restore spectra of edge pixels in the gradient domain. Finally, as the raw image is usually piece-wise smooth, we reconstruct the whole hyperspectral image by inferring from the spectra of the edges.

#### 3.2.1 *Restore the edges*

The first step is to restore the edges. But before that we need to first estimate the spatially-aligned hyperspectral image $\mathbf{i}_{\text{aligned}} \in \mathbb{R}^{XY\Lambda\times 1}$ based on the captured dispersed RGB image $\mathbf{j} \in \mathbb{R}^{XY3\times 1}$, which is equivalent to minimize $\|\mathbf{Fi}-\mathbf{j}\|_2^2$. However, this is an ill-posed problem because there's information loss in the forward model. So I use two priors for this step. One commonly used prior is that the total variation (TV) of the raw image $\|\nabla_{xy}\mathbf{i}\|_1$ is sparse. The other prior is that the edges of different wavelength images should be aligned. The spatial edges can be found by applying the spatial gradient operator $\nabla_{xy}$, and the differences in different wavelength channels can be found by applying the spectral gradient operator $\nabla_\lambda$. Because the spectra of daily objects are usually smooth and broadband, this prior indicates the spectral gradient of the spatial gradient $\|\nabla_\lambda\nabla_{xy}\mathbf{i}\|_1$ is also sparse. The two priors could be expressed as minimizing regularization terms. So the first step is to solve the following convex optimization problem:

$$\mathbf{i}_{\text{aligned}} = \arg\min_{\mathbf{i}} \|\mathbf{Fi}-\mathbf{j}\|_2^2 + \alpha_1 \|\nabla_{xy}\mathbf{i}\|_1 + \beta_1 \|\nabla_\lambda\nabla_{xy}\mathbf{i}\|_1 \tag{2}$$

We solve this problem using alternating direction method of multipliers (ADMM), which splits the original objective into 3 terms.

$$\begin{aligned} &\min_{\mathbf{i},\mathbf{z}_1,\mathbf{z}_2} \|\mathbf{Fi}-\mathbf{j}\|_2^2 + \alpha_1\|\mathbf{z}_1\|_1 + \beta_1\|\mathbf{z}_2\|_1 \\ &\text{subject to } \nabla_{xy}\mathbf{i}-\mathbf{z}_1 = 0, \nabla_\lambda\nabla_{xy}\mathbf{i}-\mathbf{z}_2 = 0. \end{aligned} \tag{3}$$

Algorithm 1 provides the details of how to use ADMM to solve this problem.

**Algorithm 1** Restore aligned image $\mathbf{i}_{\text{aligned}}$

```
1: while stopping criterium is not satisfied do
2:     $\mathbf{i}^{(k+1)} = \arg\min_{\mathbf{i}} \|\mathbf{Fi}-\mathbf{j}\|_2^2 + \frac{\rho_1}{2}\left\|\nabla_{xy}\mathbf{i}-\mathbf{z}_1^{(k)}+\mathbf{u}_1^{(k)}\right\|_2^2 + \frac{\rho_2}{2}\left\|\nabla_\lambda\nabla_{xy}\mathbf{i}-\mathbf{z}_2^{(k)}+\mathbf{u}_2^{(k)}\right\|_2^2$
3:     $\mathbf{z}_1^{(k+1)} = S_{\frac{\alpha_1}{\rho_1}}\left(\nabla_{xy}\mathbf{i}^{(k+1)}+\mathbf{u}_1^{(k)}\right)$
4:     $\mathbf{z}_2^{(k+1)} = S_{\frac{\beta_1}{\rho_2}}\left(\nabla_\lambda\nabla_{xy}\mathbf{i}^{(k+1)}+\mathbf{u}_2^{(k)}\right)$
5:     $\mathbf{u}_1^{(k+1)} = \mathbf{u}_1^{(k)}+\nabla_{xy}\mathbf{i}^{(k+1)}-\mathbf{z}_1^{(k+1)}$
6:     $\mathbf{u}_2^{(k+1)} = \mathbf{u}_2^{(k)}+\nabla_\lambda\nabla_{xy}\mathbf{i}^{(k+1)}-\mathbf{z}_2^{(k+1)}$
7: end while
```

In line 2 of Algorithm 1, the minimum of a quadratic objective is found by the conjugate gradient method. In line 3 and line 4, the slack variables $\mathbf{z}_1$ and $\mathbf{z}_2$ are updated by using the soft-thresholding operator:

$$S_\theta(\mathrm{x}) = \begin{cases} \mathrm{x}-\theta & \mathrm{x}>\theta \\ 0 & |\mathrm{x}|\leqslant\theta \\ \mathrm{x}+\theta & \mathrm{x}<-\theta \end{cases} \tag{4}$$

The first step actually provides a rough estimate of the raw hyperspectral image. However, this estimation treats interiors and edges equally. As discussed before, there are more spectral cues at the edges. So we could obtain a better estimation of the spectra when only focusing on estimating the spectra at edges.

We could find the edges by projecting the aligned hyperspectral image to the RGB color space and then applying an edge detector filter to the aligned RGB image. Because the dispersion is along the x-axis, we only care about the edges along the x-axis. I first apply a binary filter to find pixels where the spatial gradient along the x-axis is larger than a certain threshold. Then I blur the obtained image using a Gaussian low-pass filter. Finally I apply a binary filter again to find pixels where the spatial gradient along the x-axis is larger than another threshold, which provides a reliable estimate of edges along the x-axis.

### 3.2.2 Restore spectrum of edge pixels in the gradient domain

Fig. 3 has indicated that there's abundant spectral information at edges in the gradient domain. So we would like to estimate the gradient of edge pixels in this section. We use $\mathbf{g}_x \in \mathbb{R}^{XY\Lambda \times 1}$ to represent the x-axis gradient of the whole hyperspectral image, and $\mathbf{v}_x \in \mathbb{R}^{E\Lambda \times 1}$ to represent the x-axis gradient of edge pixels of the hyperspectral image. $E$ is the total number of edge pixels. $\mathbf{g}_x$ and $\mathbf{v}_x$ are linked by linear transformation operators $\mathbf{M}_f \in \mathbb{R}^{E\Lambda \times XY\Lambda}$ and $\mathbf{M}_b \in \mathbb{R}^{XY\Lambda \times E\Lambda}$, where $\mathbf{M}_f$ extracts the values at edge pixels from the full size image, while inversely $\mathbf{M}_b$ interpolates edge pixel values with zeros to restore the full size image.

$$\mathbf{v}_x = \mathbf{M}_f \mathbf{g}_x, \mathbf{g}_x = \mathbf{M}_b \mathbf{v}_x \tag{5}$$

The edge gradient reconstruction problem could be formulated as:

$$\begin{aligned} \hat{\mathbf{v}}_x = \underset{\mathbf{v}_x}{\arg\min} \left\| \mathbf{F}\mathbf{M}_b \mathbf{v}_x - \nabla_x \mathbf{j} \right\|_2^2 + \alpha_2 \left\| \nabla_\lambda \mathbf{M}_b \mathbf{v}_x \right\|_1 \\ + \beta_2 \left\| \nabla_x \mathbf{v}_x \right\|_2^2 \end{aligned} \tag{6}$$

The first term is the data term, which describes the x-axis gradient of the image formation model. The second term is the same as the third term in the previous section, which describes the spatial alignment of edges by enforcing sparse changes of gradients along the spectral dimension. The third term minimizes the second-order spatial derivative of the estimated hyperspectral image to reduce artifacts. Again, we use ADMM to solve this problem. See Algorithm 2 for details.

**Algorithm 2** Restore gradient of edges $\mathbf{v}_x$

```
while stopping criterium is not satisfied do
    $\mathbf{v}_x^{(k+1)} = \underset{\mathbf{v}_x}{\arg\min} \|\mathbf{F}\mathbf{M}_b\mathbf{v}_x - \nabla_x \mathbf{j}\|_2^2 + \beta_2 \|\nabla_x \mathbf{M}_b \mathbf{v}_x\|_2^2 + \frac{\rho}{2} \left\|\nabla_\lambda \mathbf{v}_{xy} - \mathbf{z}_3^{(k)} + \mathbf{u}_3^{(k)}\right\|_2^2$
    $\mathbf{z}_3^{(k+1)} = S_{\frac{\alpha_2}{\rho}} \left(\nabla_\lambda \mathbf{v}_x^{(k+1)} + \mathbf{u}_3^{(k)}\right)$
    $\mathbf{u}_3^{(k+1)} = \mathbf{u}_3^{(k)} + \nabla_\lambda \mathbf{v}_x^{(k+1)} - \mathbf{z}_3^{(k+1)}$
end while
```

Once we obtained the gradient at edge pixels $\mathbf{v}_x$, we can compute the gradient of the full size image using $\hat{\mathbf{g}}_x = \mathbf{M}_b \hat{\mathbf{v}}_x$.

### 3.2.3 Reconstruct hyperspectral images

After obtaining the gradient of the image at edge pixels, we can reconstruct the hyperspectral image by solving the following optimization problem.

$$\hat{\mathbf{i}} = \underset{\mathbf{i}}{\arg\min} \|\mathbf{F}\mathbf{i} - \mathbf{j}\|_2^2 + \alpha_3 \left\|\nabla_x \mathbf{i} - \hat{\mathbf{g}}_x\right\|_2^2 + \beta_3 \left\|\Delta_\lambda \mathbf{i}\right\|_2^2 . \tag{7}$$

The first term describes the dispersed image formation model. The second term uses the gradient at edges which I find in the previous step and minimizes its difference with the gradient of the reconstructed image. The parameter $\alpha_3$ describes the weight of the similarity in the gradient at the edges. In the third term, the operator $\Delta_\lambda$ computes the second-order spectral derivative. The prior is that the spectra in usual scenes should be smooth. Because this optimization problem consists only of $l_2$-norm terms, we can simply use conjugate gradient method to solve it.

## 4 Analysis, Evaluation, and Comparison to Other Methods

My implementation takes about 2 minutes to reconstruct a 256×256×31 hyperspectral image. I crop the reconstructed image into the size of 256×224×23, with 23 wavelengths from 430 nm to 650 nm in 10 nm intervals. The shrink in spatial size is because the dispersion causes marginal pixels' information lost. The short wavelengths and long wavelengths information are discarded because their relative error is high compared to wavelengths in the middle. But for many applications, the 430 nm to 650 nm range is enough as it has covered the major part of the visible band.

### 4.1 Method analysis

*Optimization parameter analysis.* There are a few optimization parameters in this method, namely $\alpha_1, \beta_1, \alpha_2, \beta_2, \alpha_3$, and $\beta_3$. Analysis shows that the results are not sensitive to $\alpha_1$ and $\beta_1$. That's reasonable because $\alpha_1$ and $\beta_1$ are used in step 1 to align the dispersed image. The aligned image is only used to detect the edge. It's not necessary to recover very accurate spectra or high spatial resolution. Thus the final results are not sensitive to $\alpha_1$ and $\beta_1$. However, the parameter choices for $\alpha_2, \beta_2, \alpha_3$, and $\beta_3$ are important. Table 1 shows sensitivity analysis results on $\alpha_2$ and $\beta_2$. I evaluate the parameter choices by comparing the reconstructed spectra and ground truth spectra on several patches in the image and computing their average root mean square relative error (RMSRE). It is defined as

$$\text{RMSRE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{\hat{x}_i - x_i}{x_i}\right)^2}, \tag{8}$$

where $\hat{x}_i$ denotes the reconstructed spectrum and $x_i$ denotes the ground truth spectrum. A small average value of RMSRE means the reconstructed spectra is close to the ground truth spectra. So the smaller the RMSRE is, the better the method is. By scanning different choices of $\alpha_2$ and $\beta_2$, we can see we get smallest average RMSRE when $\alpha_2 = 10^{-4}$, and $\beta_2 = 10^{-1}$. Using similar method, I find the best choice for $\alpha_3$ and $\beta_3$ is $\alpha_3 = 5 \times 10^{-4}$, and $\beta_3 = 2 \times 10^{-3}$.

TABLE 1
Average spectral RMSRE of recovered image by setting different $\alpha_2$ and $\beta_2$

| Average RMSRE | $\beta_2 = 10^{-2}$ | $\beta_2 = 10^{-1}$ | $\beta_2 = 1$ |
|---|---|---|---|
| $\alpha_2 = 10^{-5}$ | 0.1732 | 0.1726 | 0.1700 |
| $\alpha_2 = 10^{-4}$ | 0.1759 | 0.1686 | 0.1728 |
| $\alpha_2 = 10^{-3}$ | 0.1723 | 0.1727 | 0.1700 |

*Noise sensitivity analysis.* In this section, I analyze the influence of different noise levels on the results. This is important because the second step in the hyperspectral reconstruction algorithm is done in the gradient domain. As gradient can magnify the noise, the final result might be sensitive to the noise. In the forward model I add Gaussian noise to the captured dispersed image. The noise level is defined as the reciprocal of the signal-to-noise ratio, or rather the standard deviation of the noise divided by the average pixel value. I use PSNR of the recovered image (converted into RGB color space) and average spectral RMSRE to evaluate the model.

From Fig. 4 we can see that when the noise level is lower than 0.1, both PSNR and RMSRE almost don't change. So my hyperspectral image reconstruction is not sensitive to noise when the noise level is less than 0.1. This indicates my method is robust in practice as 0.1 is a relatively high noise level in photography and usually the actual noise level is much lower.

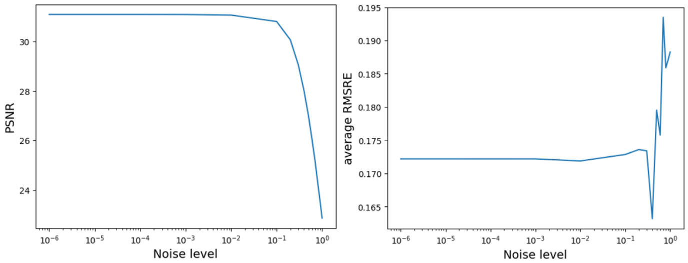


Fig. 4. The two figures show how PSNR and average spectral RMSRE change versus the noise ratio

### 4.2 Comparison to other methods

*Spatial resolution.* Compared to filter-based methods and mask-based methods, my method doesn't sacrifice spatial resolution for the spectra. In addition, the average PSNR in the results is higher than 26, indicating that the reconstructed RGB image remains spatially accurate. So this method could reconstruct high-precision spectra without harming spatial accuracy. However, if using a narrow-band filter array to reconstruct the hyperspectral image, as there are 23 wavelength channels, the effective pixel number for each wavelength channel is total pixel number divided by 23, and the spatial resolution will be low then [6]. So my method produces images with higher spatial resolution than filter-based methods and mask-based methods.

*Temporal resolution.* Because my algorithm can reconstruct the whole spectrum from a single shot, it has a very high temporal resolution. Besides, as it is not sensitive with respect to a noise level as high as 0.1, it could work well when using a short exposure time and large ISO, which is needed in shooting a high frame rate video. However, spectral scanning methods usually take tens of shots to reconstruct a hyperspectral image, which suffers from very low temporal resolution [6]. Therefore my method beats spectral scanning methods in temporal resolution.

## 5 Results

I use a public hyperspectral image dataset [11] to test my implementation. In Fig. 1 and 5, I reconstruct the hyperspectral images of lemons, a ColorChecker, and a toy dog.

*Spatial resolution.* One of the major advantages of this method compared to other methods is that it doesn't involve intrinsic reduction in the effective pixel number. From the reconstructed hyperspectral images and RGB images shown in the second and third columns of Fig. 5, we can see they all preserve very high spatial resolution. Even with a Gaussian noise level as high as $10\%$, the PSNRs of the recovered RGB images of the lemons, the ColorChecker, and the toy dog are 26.76, 30.81, and 24.39, respectively. Thus there isn't much resolution loss when recovering spectra using this method.

*Spectral Accuracy.* By comparing the ground truth spectra and reconstructed spectra shown in the left three columns in Fig. 1 and 5, we can see they match quite well. The average root mean square relative error (RMSRE) is less than 0.2. So this algorithm can produce very accurate spectra. We can easily distinguish the reconstructed spectra of the real lemon and fake lemon as there is higher intensity in the long-wavelength band in the real lemon's spectrum (see Fig. 1 (g)). Therefore such accuracy is high enough to be used in fields like material discrimination.

## 6 Limitations discussion and future work

### 6.1 Edge blurriness

By looking at the reconstructed image, we can see the edges are blurry. That's because the dispersion causes the captured image to be blurry at the edges. The edge blur in the original input image and spectral dispersion will both cause edges to be blurry in the captured RGB image. Even though the algorithm tries to align the dispersed image, it's hard to tell whether the blur is caused by dispersion or the edge of the original image is blurry in nature. That's also the major limitation on spectral accuracy because spectral dispersion can come along with edge blur and it's hard for the algorithm to separate the two factors.

In order to solve this problem, one possible solution would be to use more priors related to edge sharpness. For example, we could add another regularization term in the inverse problem to maximize the edge sharpness if we have the prior knowledge that the captured objects have clear outlines. In addition, we could sharpen the edge of the reconstructed image in a post-processing step to make the edges look clearer.

### 6.2 Depth-dependent dispersion

In this project, I arbitrarily define a spatially independent dispersion model for the image formation process. That's to say, the dispersive PSF is spatially invariant. In practice, the dispersion and PSF could be spatially dependent and it could cause a error to this method if the depth is unknown. A possible solution would be to first calibrate the camera system and store the spatially dependent PSF. Then apply a depth estimation neural network to the aligned image to analyze the depth-dependent image formation process [12]. Once the depth-dependent image formation process is obtained, we could use the same method to reconstruct the hyperspectral image.

## 7 Conclusion

In this project, I realize a hyperspectral image reconstruction algorithm with high spatial and spectral resolution from a single shot. It only needs a dispersive optic such as a

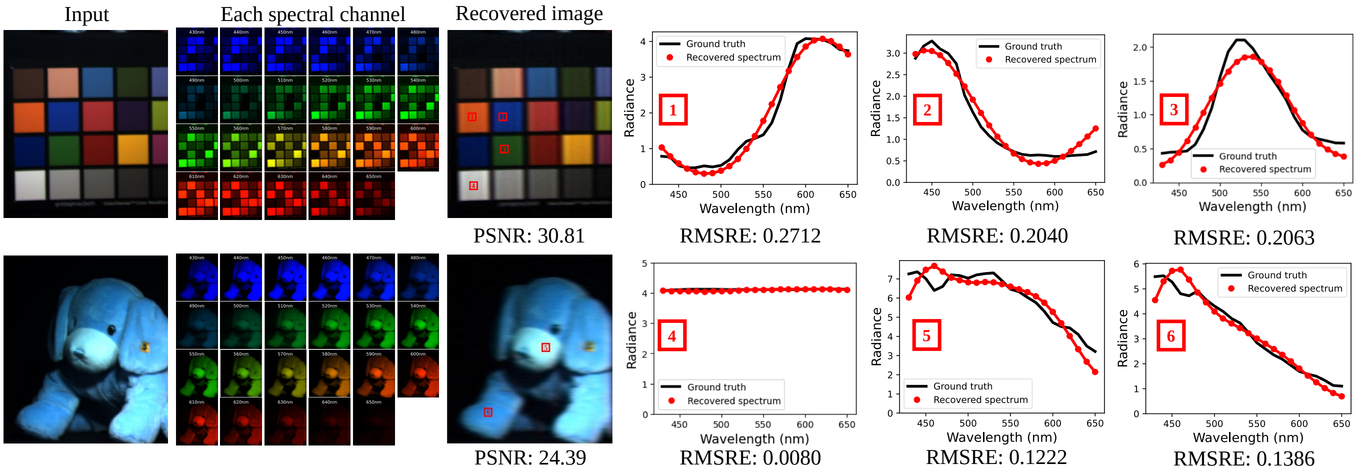


Fig. 5. Hyperspectral images reconstruction results of two images. From left to right are the RGB version of the raw images, reconstructed hyperspectral images in each wavelength channel, recovered images in RGB color space, and the comparisons of reconstructed spectra and ground truth spectra of 6 different patches in the images.

prism or a grating, and a conventional 3-channel camera. I demonstrate this algorithm on three different input scenes, i.e. lemons, a ColorChecker, and a toy dog. The reconstructed spectra show a low error compared to the ground truth spectra. The average spectral RMSRE is lower than 0.2. The PSNRs for the three scenes are 26.76, 30.81, and 24.39, respectively, indicating this method can also retain high spatial resolution. This method is not sensitive to noise when the noise level is less than 0.1, or the signal-to-noise ratio is larger than 10 dB, which is usually satisfied in most imaging systems. So this hyperspectral image reconstruction algorithm is of great practical use.

The major limitation of this algorithm is it's hard to distinguish edge blur and edge dispersion, thus resulting in blurry edges. It's interesting to explore the possibility to solve this problem by adding edge sharpness prior to the reconstruction algorithm.

## References

[1] S.-H. Baek, I. Kim, D. Gutierrez, and M. H. Kim, "Compact single-shot hyperspectral imaging using a prism," *ACM Trans. Graph.*, vol. 36, no. 6, nov 2017.

[2] M. H. Kim, "3d graphics techniques for capturing and inspecting hyperspectral appearance," in *2013 International Symposium on Ubiquitous Virtual Reality*. IEEE, 2013, pp. 15–18.

[3] X. Lin, Y. Liu, J. Wu, and Q. Dai, "Spatial-spectral encoded compressive hyperspectral imaging," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 6, pp. 1–11, 2014.

[4] A. Wagadarikar, R. John, R. Willett, and D. Brady, "Single disperser design for coded aperture snapshot spectral imaging," *Applied optics*, vol. 47, no. 10, pp. B44–B51, 2008.

[5] B. Becerir, "Color concept in textiles: A review," *Journal of Textile Engineering & Fashion Technology*, vol. 1, no. 6, pp. 1–5, 2017.

[6] N. Gat, "Imaging spectroscopy using tunable filters: a review," in *Wavelet Applications VII*, vol. 4056. International Society for Optics and Photonics, 2000, pp. 50–64.

[7] H. Lee and M. H. Kim, "Building a two-way hyperspectral imaging system with liquid crystal tunable filters," in *International Conference on Image and Signal Processing*. Springer, 2014, pp. 26–34.

[8] X. Cao, H. Du, X. Tong, Q. Dai, and S. Lin, "A prism-mask system for multispectral video acquisition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 12, pp. 2423–2435, 2011.

[9] N. A. Hagen and M. W. Kudenov, "Review of snapshot spectral imaging technologies," *Optical Engineering*, vol. 52, no. 9, p. 090901, 2013.

[10] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.

[11] F. Yasuma, T. Mitsunaga, D. Iso, and S. K. Nayar, "Generalized assorted pixel camera: Postcapture control of resolution, dynamic range, and spectrum," *IEEE Transactions on Image Processing*, vol. 19, no. 9, pp. 2241–2253, 2010.

[12] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2002–2011.