Hello, everyone. My name is Jiayu Huang. Today I'm gonna to introduce my master thesis to you. The topic of my thesis is 'Short-Term Illumination Change Detection and Compensation in Video Sequences'.

I will present the following five parts in detail. First I will talk about the motivation of the thesis. After that, it's the introduction of my thesis. Then I'm going to show you what I have done, which method I have implemented and what I have got so far. There are also some unsolved problems, I wish I could get some advices from you. Last but not least, I will talk about the next steps in the future.

So let me show you the motivation of my thesis. First, let us watch a video sequence which has lots of flashes.

In this video, we can find multiple flashes. More flashes means more difficult to compress the video. It consumes more because of more complex illumination compensation schemes. So what effect does the flashes have on each frame? For an image, the intensity is formulated by illumination multiplying reflectance. As we can see, illumination is the light irradiating on the object. And object surfaces have different reflectance. For people's perception, **Objects** appear different colours because they absorb some colours (wavelengths) and reflected or transmit other colours. And the brightness of objects is different from each other because of reflectance variations. Even with same illumination change, different objects have diversity luminance variations corresponding to different surfaces reflectance.

I will describe some basic concepts in the next slide. There are some concepts related to flash detection, illumination compensation and motion estimation. ~~I will introduce the concepts in flash detection and some basic definitions of this part. For illumination compensation, some algorithms are introduced. Lastly, motion estimation is presented in detail.~~

Let's take a look at the flowchart first.

Flash affects the luminance and chrominance characteristics of the scene abruptly and locally only to a few adjacent frames across the video sequence. The flash is mainly detected based on the luminance difference between current frame and previous frame. Also, we can detect flash bi-directionally by considering the luminance of three consecutive frames. And determine whether the current frame has flash

or not. After I detected where the flash was, I generate a ground truth for flash. And separate them into two categories, single flash and consecutive flashes.

As we can see in the figure, Y-axis is mean luminance of frames, and the X-axis is the frame index. The peaks of the plot represent the flashed frames of the video sequence. The red dots indicate the single flash frames and the green triangles indicate the consecutive flashes frames. I also plot the cdf of the luminance of frames. As we can see in the figure, the blue and higher ones stand for frame without flash. The lower red lines indicate frames with flash. It's an auxiliary for flash detection. For frame with very explicit flash, it's easy to find the index of frames. However, some frames have very slight flash in very small area. So it's a challenge to detect whether the frame has flash.

So after I have the information of flash position, I can compensate the frames. There are two main methods for illumination compensation. Linear compensation and nonlinear one, such as gamma correction. Because the flash is very unique illumination, so I apply an adaptive linear pixel wise illumination compensation algorithm. Based on the previous frame, I calculate the difference between these two frames in three components: Y, U ,V. And set thresholds for these 3 components to determine which pixel should be compensated. Based on the conversion from YUV to RGB, I can get a frame which has been assigned new pixel values. There is an example of single flash frame. The first one is flashed frame, the second one is the compensated frame. But due to motion, I can't get a precise compensated frame at the edge of flash.

For motion estimation, I choose optical flow as a basic method. Because I want to have a pixel wise motion vector. I implement dense optical flow. There are some classic methods of optical flow. Farneback is a good one to estimate the motion. First, the method approximates the windows of image frames by quadratic polynomials through polynomial expansion transform. Second, by observing how the polynomial transforms under translation (motion), a method to estimate displacement fields from polynomial expansion coefficients is defined. After a series of refinements, dense optical flow is computed. The Farneback algorithm generates an image pyramid, where each level has a lower resolution compared to the previous level. When you select a pyramid level greater than 1, the algorithm can track the points at multiple levels of resolution, starting at the lowest level. Increasing the

number of pyramid levels enables the algorithm to handle larger displacements of points between frames. However, the number of computations also increases. The diagram shows an image pyramid with three levels. The tracking begins in the lowest resolution level, and continues until convergence. The point locations detected at a level are propagated as key points for the succeeding level. In this way, the algorithm refines the tracking with each level. The pyramid decomposition enables the algorithm to handle large pixel motions, which can be distances greater than the neighborhood size.

The following slides show the results of different optical flow algorithms.

We write the “classical” optical flow objective function in its spatially discrete form as shown in the slide
where u and v are the horizontal and vertical components of the optical flow field to be estimated from images I_1 and I_2, λ is a regularization parameter, and ρ_D and ρ_S are the data and spatial penalty functions. First uses a non-linear pre-filtering of the images to reduce the influence of illumination changes. Then uses median filtering during optimization to *denoise* the flow field. ~~Besides the Farneback algorithm, I also implemented some modified optical flow. Examining the non-local term suggests a solution.~~ For a given pixel, if we know which other pixels in the area belong to the same surface, we can weight them more highly. The modification to the objective function is achieved by introducing a weighted non-local term.

Besides adding a non-local term to the objective function, it can also use a more robust penalty function for optimization. Lorentzian is a good choice to apply. Given a robust formulation, there are numerous optimization techniques that can be employed to recover the motion estimates and the most appropriate technique will depend on the particular formulation and choice of r- function. I implement the GNC algorithm for optimization. It looks better than before. But there are also some mistake estimation at some regions. Yet not so good to be a promising algorithms. Because we need a very precise motion estimation to compensate it.

Optical flow has been commonly defined as the apparent motion of image brightness patterns in an image sequence. While one of the primary assumptions of optical flow is illumination constancy. ~~So it interprets optical flow merely as a geometric transformation field.~~ The

standard *data conservation constraint* assumes that the image brightness of a region remains constant while its location may change. This assumption underlies the common correlation and gradient-based approaches and is violated when a region contains illumination change. In case of flashed video sequence, the new definition is a complete representation of geometric and radiometric variations in dynamic imagery. This is more consistent with the common interpretation of optical flow induced by various scene events. One way to deal with large illumination variation is to replace the assumption of constancy of pixel brightness values with a more realistic model. For example, Gennert and Negahdar- ipour [15] have assumed that the brightness at time t+dt is related to the brightness at time t through a set of parameters that can be estimated from the image sequence. This definition extends the 2D motion field vector (u, v) to a 3D transformation field given by (u, v, de). The last component, describes the radiometric transformation of the image sequence. It replaced the brightness constancy constraint of Eq. (1) by the following more general form in which the brightness at a pixel in two consecutive images is related via the motion parameters u and v and the radiometric parameters M and C, as shown in the slide.

Besides classic methods, there are also deep learning methods which I have not implemented until now.

The results I have got is implementation based on the best performance algorithm. The first one is illumination before motion. The red circles mark the not so precise motion estimation. The action looks like somewhere between the two frames. Especially at the gestures of people's hangs and feet.

Currently, the problem I meet is mainly the accuracy of motion estimation.

The above are the problems I encountered. For the future work, I will continue to find a more precise motion estimation algorithm. Maybe deep learning method is a promising direction.
Apart from this, I will find a better illumination compensation method, such as adaptive local gamma correction.