Notes for PyData 2018 talk.

Slide 1: Presentation

- Presentation, mention my talk last year about Shearlab.jl and mention that this is a good application of this library.

Slide 2: Main goal.

- The main goal of this talk (and therefore of this project) is to present a novel technique of depth map reconstruction of a scene from a limited number of views (the depth map contains the depth of most of the points in the scene with respect to the camera position), this can be applied in view synthesis and rendering for free viewpoint VR (where you can explore the image in depth).
- Explain the main building blocks of the technique: Light Field and Shearlets (the thing that I talked about last PyData).
- Show a free hardware/software implementation using julia, python and Raspberry Pi.

Slide 3: What is a Light Field

- Light can be interpreted as a field, i.e. assignment of a vector to each point in the space.
- Propagation of light rays in the 3D space is completely described by a 7D function, called the plenoptic function, that describes the intensity of the light field at a 3D point (x,y,z), with angles theta and phi, wavelenght lambda and period tau (show the image of the plenoptic function.
- The plenoptic function can be simplified to a 4D light field or just Light Field which quantifies the intensity of static and monochromatic light rays propagating in half space.
- The main idea behind this is that when you take a picture you just have the projection of the light field on the camera plane, therfore you dont have any directional information of the light field, the directinal information has all the depth map encoded.

Slide 4: 4D Light Field Representation

- There are different parametrizations to represent this 4D light field:
  - A point in a 2D surface and 2 angles.

  - Two points in a sphere.
  - Two points in parallel planes, the so called 2 parallel plane parametrization, we will center on this.

Slide 5: From LF to 3D

- It is still unclear how one can extract the 3D information from the 2 plane parametrization of the light field. That is why I am going to introduce the next three concepts.
- Stereo Vision (you might already heard of this), this is how we as humans see in 3D (or what we think is 3D), the human brain generates the 3D depth perceptio of its sorroundings by triangulating the points of a scene using the information coming from both eyes. In this case, the image planes in each eye correspond to the parallel planes in the 4D light field representation.
- Epipolar Geometry, is the generalization of stereo vision with more than two viws, assuming the epipolar constraint.
- The epipolar constraint, assumes the knowledge of the camera motion and analyses the object position.
- Show the image with the planes, if you have 2 image planes, one image point in the scene, the epipolar plane is the plane spanned by the point and the projections of the point in the image planes (projected in the line that connects to the lens centers), the epipolar line is the lines that of intersection of the epipolar plane and the image planes.

Slide 6: Epipolar Plane images on Straight Line trajectories.

- In this case we are assuming that we have N different views (pictures) taken with a camera or a set of cameras that is movien in time on an straight line perpendicular to its lens axis.
- If we keep our attention on all the feature points that lie in an horizontal line the will describe a set of lines on the plane x vs. time. This is called the epipolar plane image and can be interpreted as 2D slides of the 4D light field.

Slide 7: Depth map estimation

- By just looking at the diagram, where p is the feature oint in the scene, and u1, u2 are the positions of this points in each image plane one can compute the relative distance of the point and the camera (the depth) by the next formula, therefore the depth is linearly propotional to the slope of the corresponding line in the EPI.

- In order to compute the slope with any line detecting algorithm we need very high sampling rate (a lot of images from the scene taken), Nyquist criterion give us a lower bound for the sampling rate: At least a maximum of 1 pixel disparity between nearby views to avoid aliasing and other artifacts.
- In this work we used compressed sensing theory and sparse recovery theory to increase the efficiency of the algorithm by reconstructing the light field with sub-Nyquist sampling rates. I will explain that in the following.

Slide 8: Commercial LF (Epipolar) cameras

- There already exists at least two commercial epipolar (light field) cameras Lytro and Raytrix, that uses a lens array to take all the different needed views for the epipolar image processing.
- The withdraw is that they are very expensive (order of 500 - 3000 euros), slow, and far from being open source, the related research articles are very obscure for industrial purposes. Even though they still a cool product, and for this reason google just acquired the whole Lytro company last year.

Slide 9: Our approach: Sub-Nyquist reconstruction via inpainting

- As I mentioned to be able to compute the depth map by analysing the 2D slides of the 4D light field called, EPIs one has a lower bound of sampling rate. For high resolution images this lower bound is quite high, and the number of images required of the scene will be enormous.
- One can use sparse recovery theory (based on compressed sensing and harmonic analysis) to be able to decrease the Nyquist sampling rate, and therefore the computational complexity, but still be able to reconstruct the light field.
- The idea is to sample a sparse set of contiguous views (pair by pair). The upper image show the obtained raw EPI when one does that. One cannot recognize lines there, if one separates the pair of contiguous views accordingly, one can recognize some lines but a typical line detecto wont be able to detect them, therefore one needs to fill up the space in the middle.
- This action of filling wholes in images is called image inpainting and is well known inverse problem. One can use the machinery of sparse reconstruction to solve it, and that is where the Shearlets arrive.

Slide 10: (General) Image inpainting.

- The mathematical formulation of inpainting can be read as recover an image (that can be interpreted as a function in some space) from incomplete measurements.
- This is typically applied to old images with lost parts or to incomplete seismic data.

Slide 11: How to inpaint?

- The theory that study the recovery based on incomplete measurements is compressed sensing, it uses strongly a concept of frame and sparsity.
- A frame of an space is the generalization of an orthonormal basis (it includes, redundancy that allows sparse representations).
- One can then recover an image from incomplete measurements by minimizing the ell1 norm of the coefficients within the frame. The quality of the reconstruction will depend of how much the frame sparsifies.
- Last year I talked about the Shearlet system and related transform and how I implemented it in julia, I used this library (Shearlab.jl), one can see the Shearlet system as a multidimensional generalization of the Wavelet system and transform.

Slide 12: Followed Pipeline

- The end-to-end pipeline for the Light Field reconstruction is the following:
  - Acquisition of rectified images sequence, of the scene, as I said we acquired a set of sparse pairs of consecutive views.
  - After acquiring the views we need to track the points to be able to paint the Epipolar Plane images. Of course this is not an easy task. Corners are good points to track, we used the Shi-Tomasi algorithm with OpenCV in python to detect the corners to track.
  - Once we have the corners, we used the Lucas-Kanade algorithm for flow tracking to track the points in the different views, we obtain a set of sparse Epipolar Planes for each horizontal line on the images plane.
  - We then inpaint using shearlets with Shearlab.jl and julia.
  - Once we have the inpainted EPIs we used the Hough Line Transform to detect the lines at the EPI and then compute the depth map using the corresponding line slopes.
- The library LightFields.jl provide functions that do each of this steps, one can go to the github repository and check also the examples.

- I used julia for everything but the computer vision tasks (corner detection, point tracking and line detection), for which I used the python api of OpenCV, since I could not make this work with the julia OpenCV api.
- Those are the technical details of the camera, given by the data provider.

Slide 13: Used Data Set, Church

- The data set consists of pictures of a church in Zurich, this are the first and the last image of the sequence.

Slide 14: Point Tracking Results

- Using the Shi-Tomasi algorithm we found 336 strong corners, and using the Lucas-Kanade algorithm we track them along the scenes. (Show the leaved traces).

Slide 15: Particular EPI Example

- Show the example of an EPI, with the horizontal strip of points in the EPI, corresponding to different features. Show the continuous EPIs that uses all the views, and the sparse that we used, that uses a 7th of views.

Slide 16: Results on EPIs inpainting

- Show the example of the inpainted EPI.

Slide 17: Results on the line detection and depth map estimation

- We used the Hough line detector to compute the depth map for each point using the provided equation.
- Show the correspoinding depth map.

Slide 18: Open Hardware implementation

- Show the picture and the hardware. Mention that the camera is not working. Also mention that the Memory of the Raspberry PIs is not sufficient to compute the shearlet system but it is possible to compute the corresponding transform

Slide 19: Future work

- The next step is to have a more stable version on the Raspberry pi, and also do Light Field rendering that uses the depth map to render a 3D graphic version of the scene.

Slide 20: Thanks

- Any questions