Machine Translated by Google

Computer vision - 2015

Homework 2

9/5/2023

General instructions:

- Submission is in pairs or individuals only.
- In this exercise, each question must be implemented in the Matlab or Python language, when existing libraries can be used for operations Basics such as reading and saving images, drawing on an image and more specific functions that are allowed to be used in them according to the details in each question. Recommended Matlab libraries are: -Computer and Processing-Image Vision Toolboxes as well as the feat_vl library. For Python, you can specify -scikit, OpenCV, Pillow .image
- You must submit all the code you wrote\n (not including code from other libraries you used\n) in a single file .code.zip in the name
- Submit all the results that you will be asked to show when there is a separate folder for each stereo problem, with everything packed in a single file named zip.results
- One document in pdf format must be submitted, in which name and ID must be specified. The bonus explanations should be there (if you answer this section). • The exercise must be submitted through the model website. You can submit updates of the submission (as long as it is before the deadline The submission (as well as appeals for review in case of need in a repeated email in the same thread.
- The exercise to be submitted by Wednesday, May 31 at 11:59 p.m. Each day late will result in a reduction of 4 points, until for the last date of Friday, June 2. No postponement will be granted (except for justified requests that arrive by May 27).

Machine Translated by Google

Questions:

Photometric Stereo based View Synthesis (100 points). 1

The goal is to create a series of new viewing angles of a scene, linking a pair of images in rectified mode. This can be done by changing the pose of the camera and synthesizing the image that would have been acquired from the new shooting angle. In each example there is a pair of images in rectified mode taken by a pair of identical cameras (with the same intrinsics) when the distance between the cameras (baseline) is about 10 seconds m (0.1 meter).

Steps:

A. Compute the disparity maps (from both directions) by the following local stereo algorithm. The all-takes-winner method must be implemented after aggregation on volume-cost calculated on the basis of consistency: filtering with census-transform

- Calculate transform-census on each of the images
- Calculate the volume-cost (note that the maximum disparity is given in the txt file).
- Perform local aggregation
- Find the minimum for each position
- Filter matches based on test-consistency

B. Calculate the two depth maps from it.

third. For the left image: calculate a reprojection of all image coordinates (pixels) to The three-dimensional space. This can be done using the matrix K of the intrinsics (the inverse) and the map The depths D. The points in 3D will be in the coordinate system of the camera, meaning that we use

. 0 = , |] = when = the camera matrix [

d. Drop the 3D points back onto the left camera plane, in its original position. now, Synthesize the resulting image by copying the pixel values in RGB from the original image. The result should be the same as the original image (to the point of holes) - this is in total a correctness check of 2D-3D-2D process

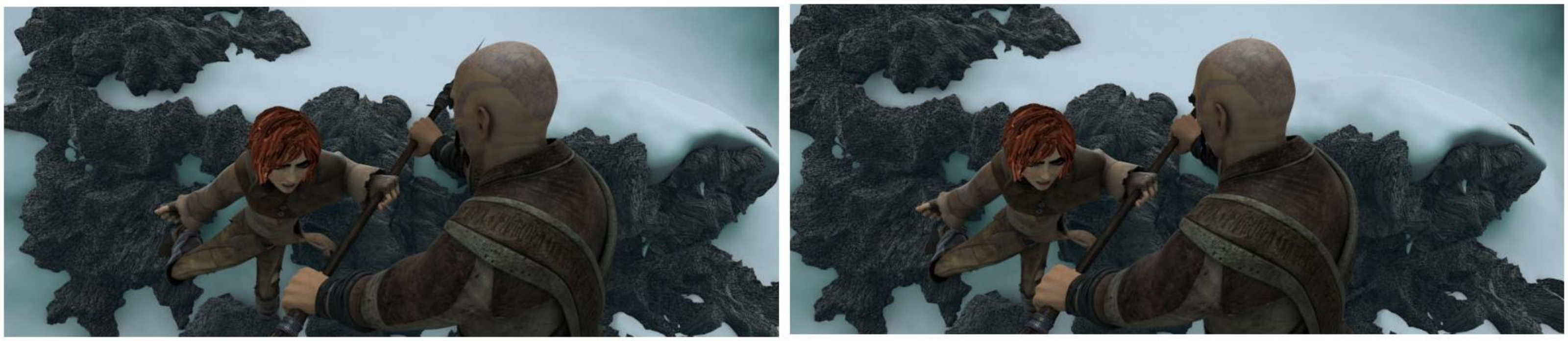
God. Repeat for a series of 11 camera positions, at uniform 1 cm intervals on the baseline between the two images. This can be done by manipulating T in the extrinsic matrix.

Note: We have provided you with a solved example, from which the visualizations below were taken. Among other things, there are txt files that can be loaded of the disparities and depth matrices (in addition to their visualizations). You can start solving directly the second part of the exercise (steps c and onwards) given the depth maps (at least until we learn about Stereo calculations in the next lesson.

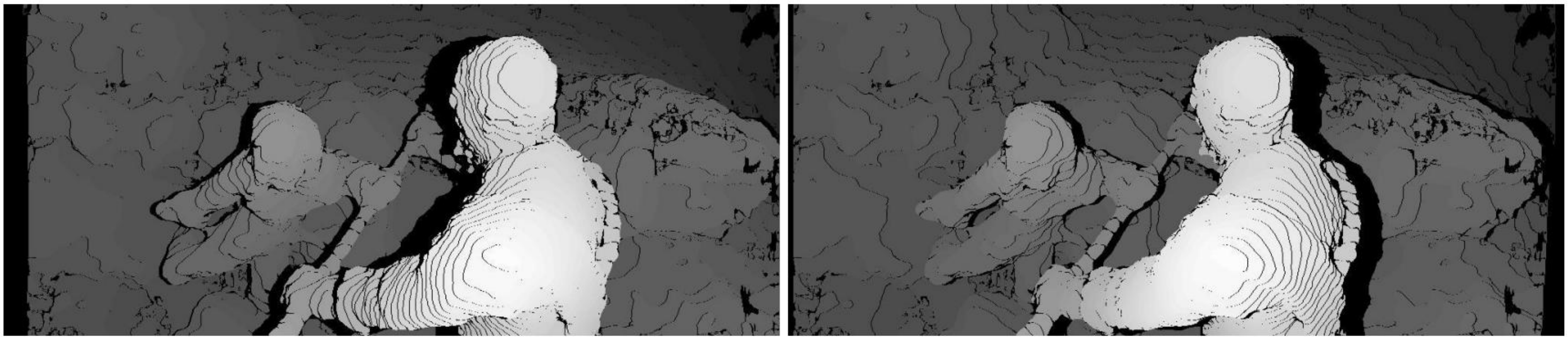
Machine Translated by Google

To submit (for each scene, in a separate folder):

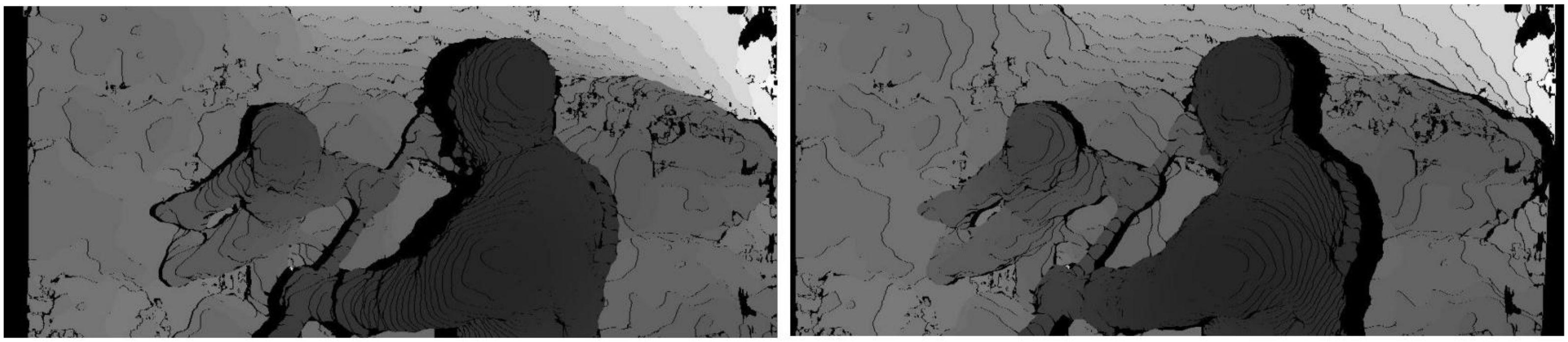
: im_left.jpg, im_right.jpg The names with the originality of the pictures pair 1



2. A pair of disparity images with the names jpg.right_disp, jpg.left_disp (in the areas without the data you need to put zero and finally divide the entire image by the maximum value to get an 'image' in [0,1]:

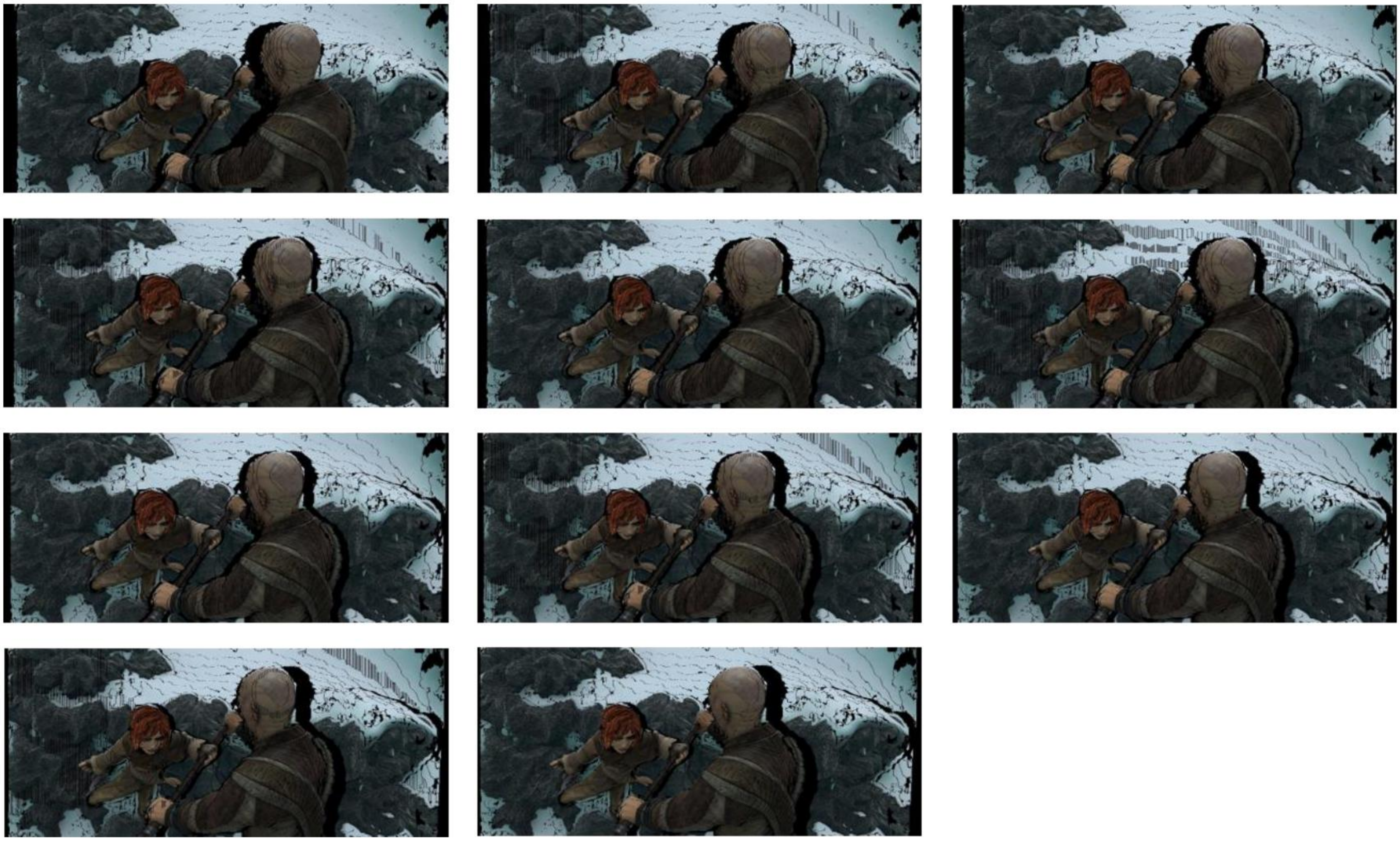


3. The pair of depth images with the names jpg.right_depth, jpg.left_depth (also here, in the areas without the data you need to put zero and finally divide the entire image by the maximum value to get an 'image' in [0,1]:



Machine Translated by Google

4. The series of synthesized images with the names jpg.i_synth(,11...1,=i with the rest of i increasing from left to right):



And a bonus (10 points): pay attention to the "holes" (blackened areas) in the results. There are two types of holes: (I) (lines thin; (II) (more significant holes (that look like silhouettes). a. Explain the source of each of these two types of holes b. Propose and explain a solution in which there are no holes at all. It is worth remembering that both source images can be used In this case.

third. Create another new folder with all the above outputs, also for this case.