Instructions for running Motion-tracking application

Ashton Krajnovich

Dr. Shiling Pei

Colorado School of Mines

Workflow

1. Import video and read to frames
2. Convert frames to B/W
3. Detect bright areas in each frame (create masked images)
4. Initialize locations of each marker in the first frame
5. Track marker locations from frame-frame, recording displacements
6. Convert measured displacements from pixel to physical coordinates

    Result: Time-series plot of displacement for each marker

Physical Setup

- Mount video camera on tripod, ensuring the camera is facing perpendicular to the direction of motion of the shake table
  - (Tripod not included)
- Place markers as needed (using Play-dough to fix positions)
- Measure physical distance between two markers and record
- Camera is set to use .MP4 file format, 60 FPS, 1080P

Items Provided

- Point-tracking script: *tracking.m* (MatLab application)
- Video Camera (Panasonic HC-V180), LED lights, Play-dough
- Example shake-table videos:
  - S1010001.MP4 (0.5 Hz, 5 in.)
  - S1010002.MP4 (1 Hz, 1 in.)
  - S1010003.MP4 (2.5 Hz, 0.5 in.)
  - S1010004.MP4 (4 Hz, 0.25 in.)
  - S1010005.MP4 (Variable frequency/displacement)
- JPG frames for each of the example videos (for convenience)
  - S101000*.MP4_xxx.jpg (xxx – frame number, * - video number)

Initializing

- Navigate your working directory to the *Deliverables* folder
  - Include any new videos that will be processed in the same folder
    - Example videos S101000*.MP4 included

Run *tracking.m* and follow the on-screen directions:

1) `Do you need to read in the video file? (Type Y for yes, anything else for No):`

Type ‘Y’ if you need to read a new video to frames.

2) `Enter the filename, with extension, for the video to process:`

If your video is already read to frames, still include the proper video name here. It is used for identifying corresponding frames to process.

Initializing cont.

3)
```
Now the frames will be transformed to grayscale, and the LED markers detected in each frame.
Press ENTER to continue: >>
```

This section of the code will identify bright regions in each frame and create binary masks (of bright pixels). May take a while, so monitor the progress bar.

4) Now you will initialize the marker locations in the first frame – user input is required for this step

Specifically, you will be using the data cursor tool to identify the locations of each marker from the binary mask of the first frame

Initializing Marker Locations

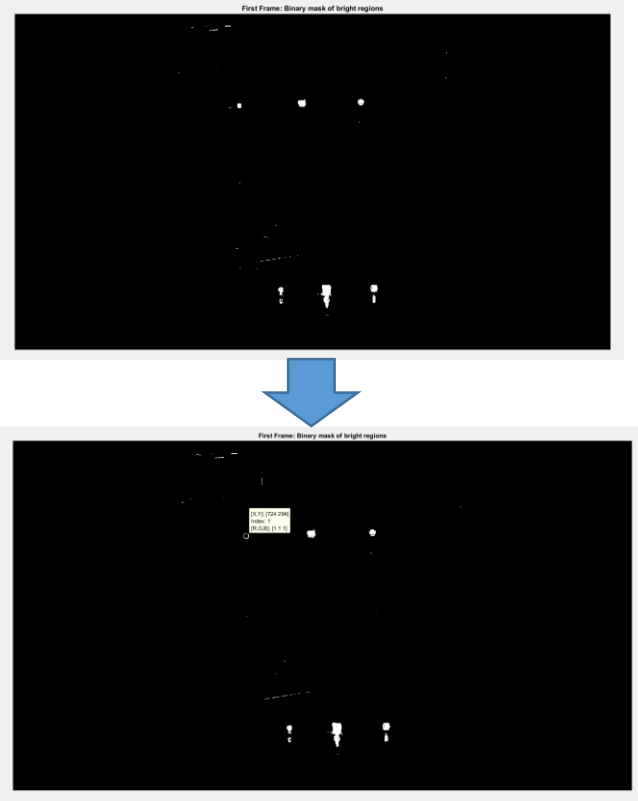
5)

```
Now you will begin demarcating regions containing each marker using the masked first frame.
Press ENTER to continue: >>
How many markers are present in the image?: >> 6
You will use the data cursor tool to manually examine data values, and select the initial
locations for each marker region.
From the following image, select approximate marker locations for each marker, one by one.
Use alt-click to select each marker location, then right click and
save the location using Export to Workspace.
Save as a variable "markers".
Press ENTER to begin:
```

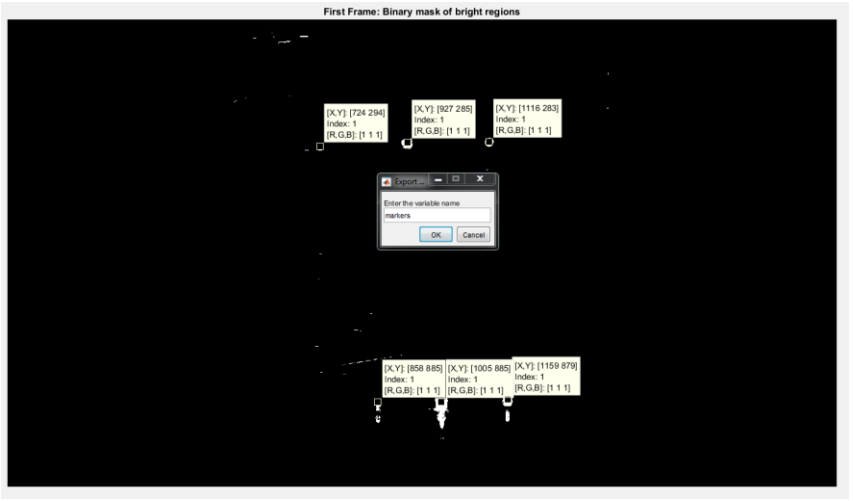
Following on-screen directions, input the number of markers used. Following the prompt, a screen will pop-up displaying the binary mask of the first frame. Use Alt-Click to select the marker locations.

Initializing Marker Locations

6) Use Alt-Click to select the marker locations from the first frame.



Initial locations will be stored as a variable markers, where markers(i).Position will be the (x,y) position of marker i.
N.B.: The indexing of markers(i) will be such that markers(1) is the last marker that was selected and markers(N) is the first marker selected (where N is the total number of markers).



Point Tracking

7)
```
Point tracking will now commence using binary masks and initial marker locations.
Press ENTER to continue:
```

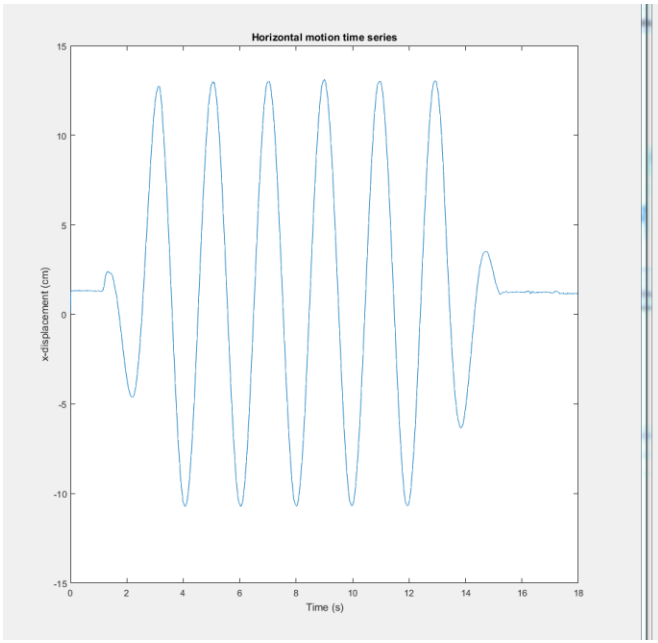
Once markers are selected, continue with on-screen directions. Point tracking will commence – this section takes a while so monitor the progress bar and grab a cup of coffee..

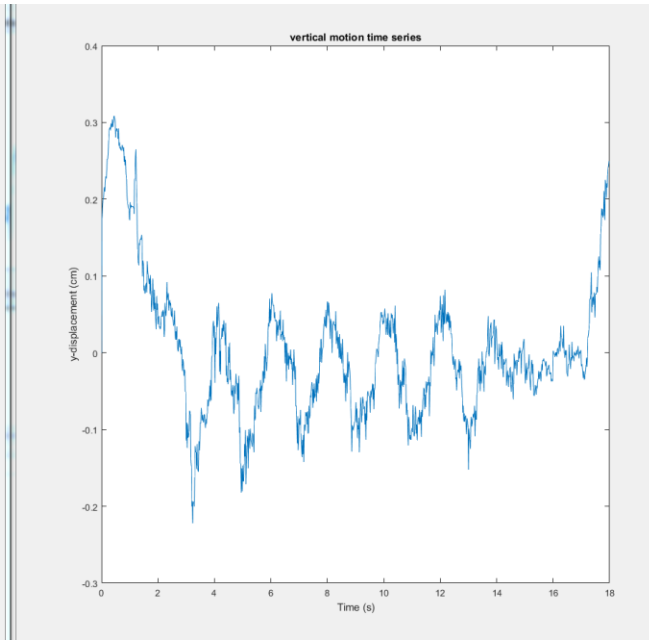
8)
```
Now spatial calibration will be applied to the frames to convert pixel distances
to physical distances.
You will be prompted for the two markers between which you measured the physical distance.
Press ENTER to continue: >>
Enter the number of the first marker (using the order you selected the markers): >> 1
Enter the number of the second marker (using the order you selected the markers): 5
Enter the physical units used: cm
Enter the measured physical distance between the two selected markers: 10
```

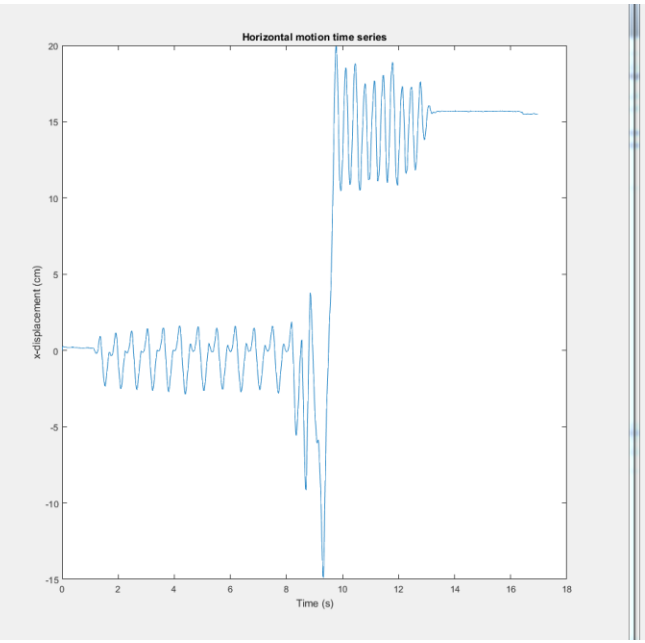
Enter the distance measured between the two selected markers for spatial calibration. Take careful notice of the odd convention for marker indexing (mentioned on the previous slide)

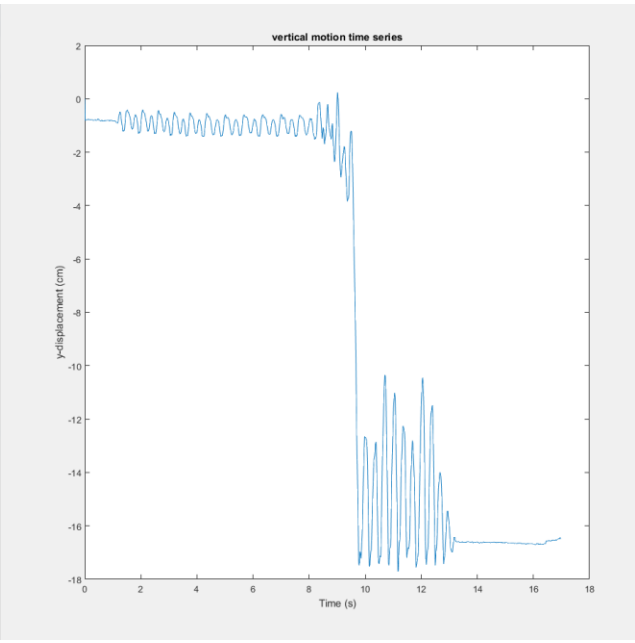
Results

End result will be time-series plots of x- and y- displacement, for each marker. (example: Left (S1010001.MP4), Right (S1010005.MP4))









Troubleshooting

- A key step to this process is spatial calibration. **Make sure you measure a physical distance (in a plane) btw two markers, and record which markers the measurement is between.**
- If the binary masks don't appear to be capturing the markers well, adjust the variable brightThresh to be a smaller value (default set to 253). Reducing the value will result in increased brightness sensitivity but also increased noise levels.
- Reading videos to frames is time consuming. I recommend doing this for all videos first.
- It is unlikely to have problems, but if the point tracking is 'losing' a point between frames, consider increasing the value of the variable searchRadius (radius of pixels to search for a marker in the next frame, default = 100).
- As with any debugging, try re-running the code with a fresh start if any weird errors are occurring. *If all else fails, contact me (Ashton): akrajnov@mines.edu*