Computational Vision

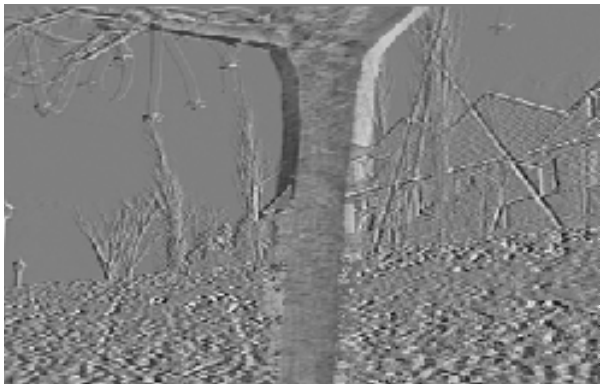
Laboratory for motion

Miquel Perelló Nieto
Marc Albert Garcia Gonzalo

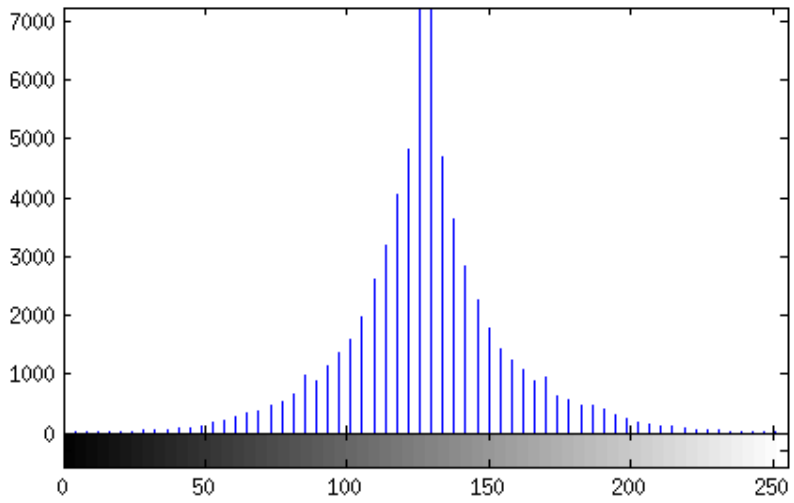
1.1 Introduction

Question: Which information appears in this difference image ? What kind of information does us the white and black pixels give ? And what kind of information does us the gray pixels give ? Comment your response.

Difference image:



Histogram of the difference image:



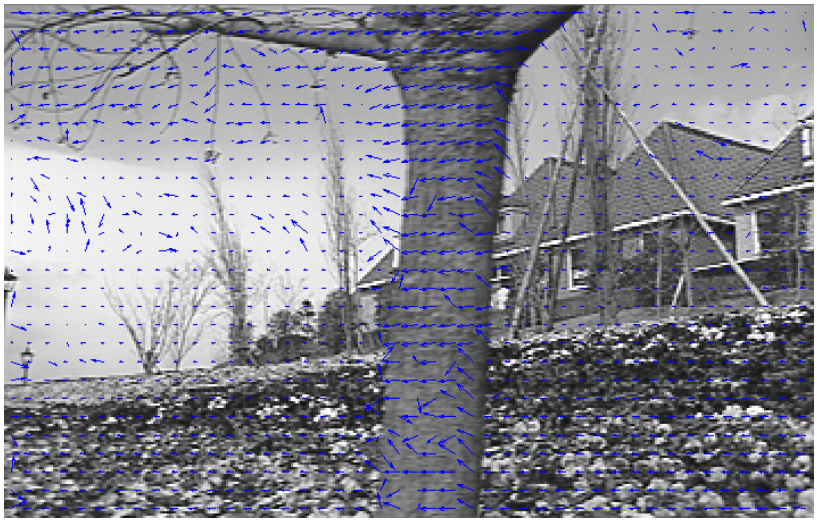
The gray colors around 128 indicate the absence of motion. This can be produced because the the image did not move. Also, it is possible that the image moved, but because it moved in an area where there is no texture, motion is not perceived in the image.

Brighter and darker areas on the difference picture show changes in the image. In this series of pictures the flow of motion. Bright areas represent the transition from bright to darker colors, while dark areas represent transitions from dark to bright colors.

> *Question: Can you see any effect due to the lack of gradient ? Can you see any effect due to an occlusion ? Where are these effects produced ? Why ? Please comment your response.*

It is possible to see the lack of gradient in points that do not have any direction. The occlusions often makes to appear opposite narrows, this can be because the principal object can be interpreted as moving in one direction, and the occluded object appear to move in the opposite direction (or near directions).

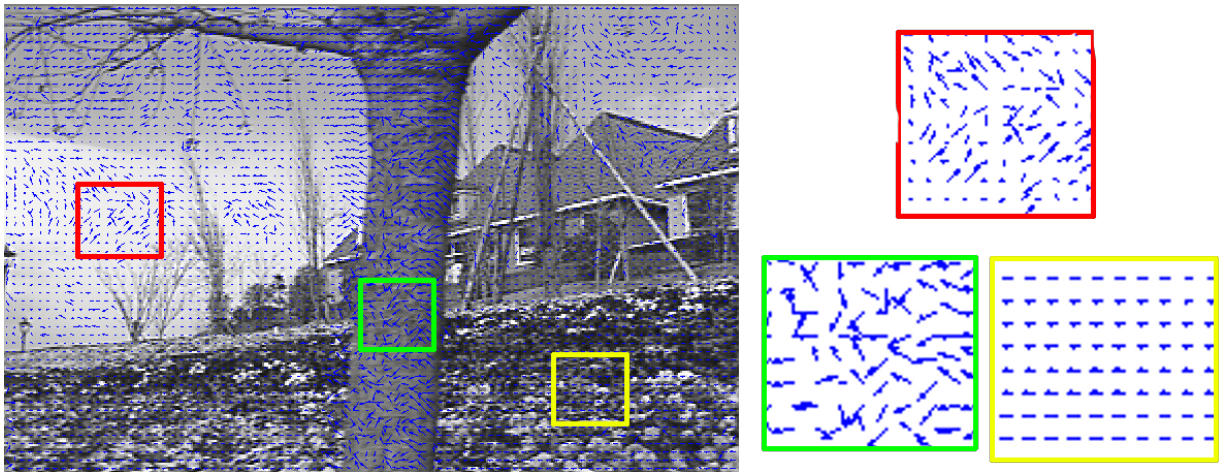
8x8 block size:



1.2 Effect of the size of the block

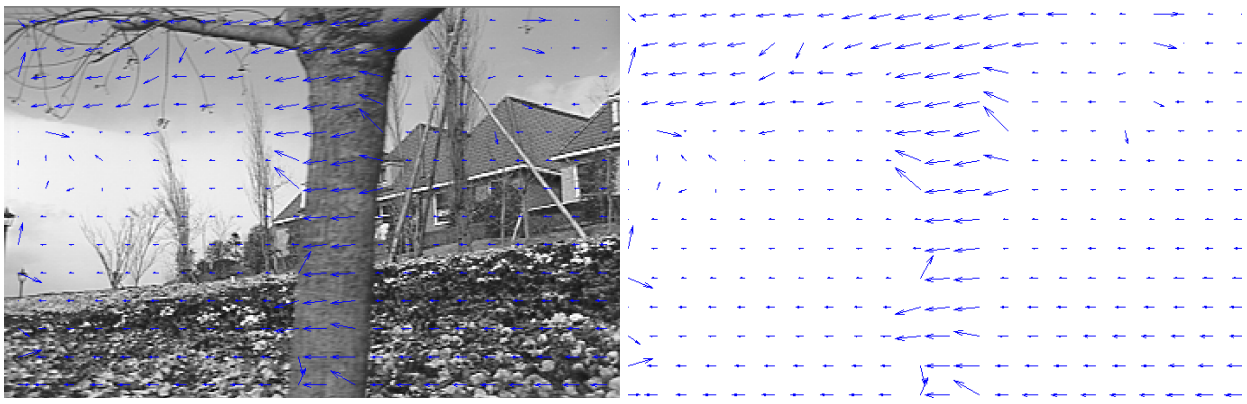
Question: Comment on the obtained results with each block size. Do you see again any effects of lack of sufficient gradient or occlusion ? Do you think that there is a best block size ? Comment your response.

4x4 block size:



In this case we can see how the direction of the arrows does not follow the image movement in most of the image. This is specially true where there is more lack of gradient. In places where there is a higher and smaller level of detail like the floor, the arrows look to detect better the movement.

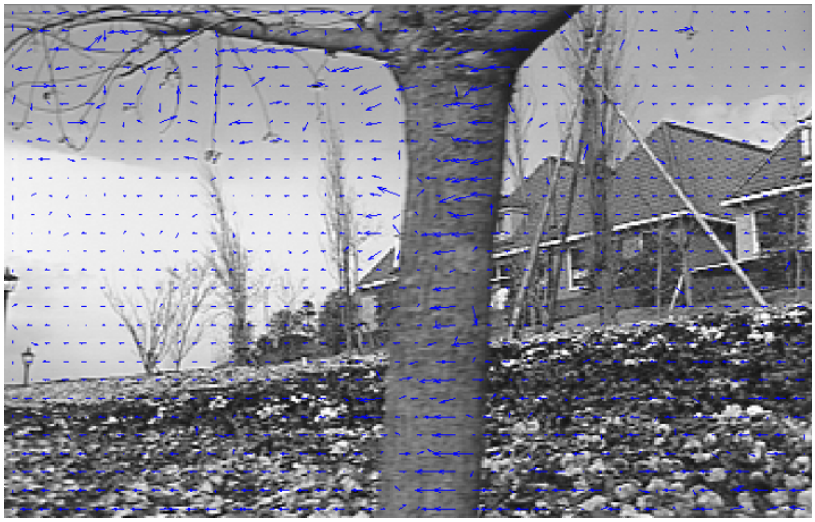
16x16 block size:



In this case we can see how the movement is predicted quite well, specially for the tree. With bigger block size, the motion of big objects is detected better. There are some exceptions, where the motion is detected in the opposite direction, but in general the results are good. We can see how the the results for small objects (like the flowers) are also predicted quite well. So, we can conclude that for this image, the block size of 16x16 is better than the other ones tried.

1.3 Effect of the search algorithm

Question: Note that the time needed to estimate the motion vector is much lower than for the full search ? Why ? Comment your response.

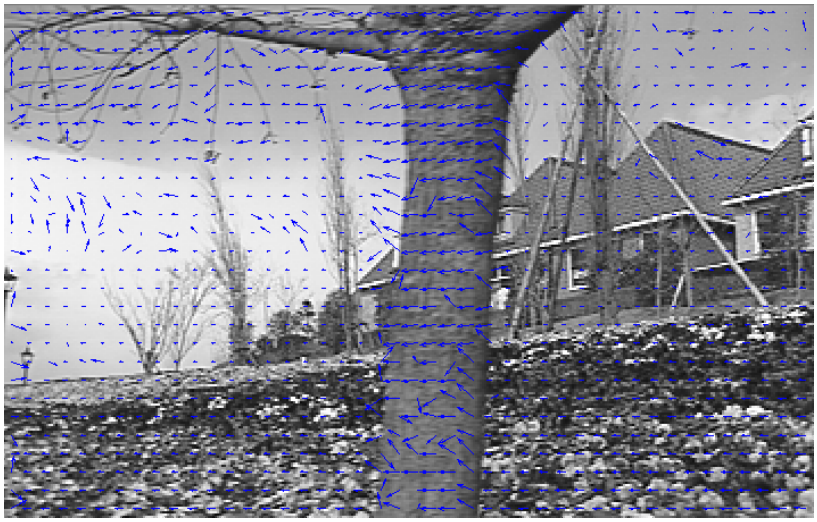


This is because the orthogonal search algorithm looks does not look for the global best solution, but for a solution which is the best among a reduced set of solutions. This set of solutions is based on the difference among the values of the pixels found orthogonal to the pixel we are looking the motion for.

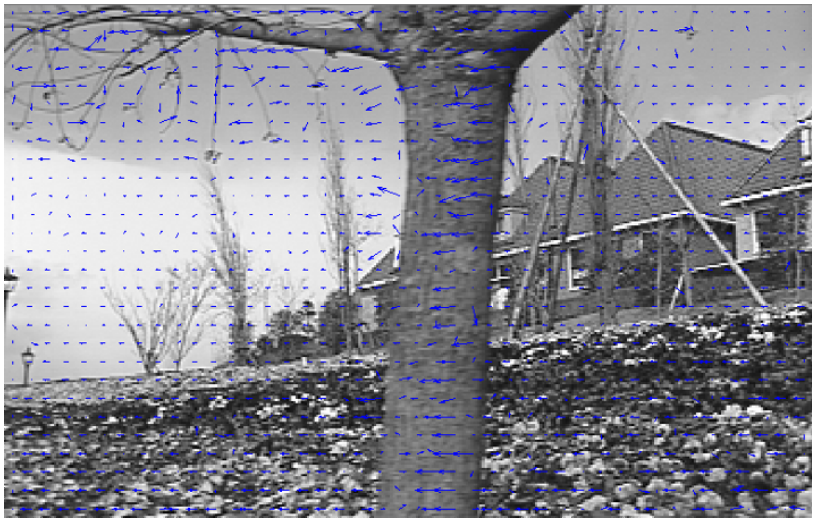
As the comparisons made are much lower, the computational cost, and the elapsed time is much lower using the orthogonal search algorithm.

Question: Why are (sometimes) the vectors different ? Which of both methods do you think allows to obtain the best results ? Comment your response.

Brute force:



Orthogonal search algorithm:



The results are the same only when the local best solution found by the orthogonal search algorithm is the same as the global best solution, but this is not guaranteed. In many cases it is possible that a the solution found is not the best possible, but the best one in the search space of the algorithm.

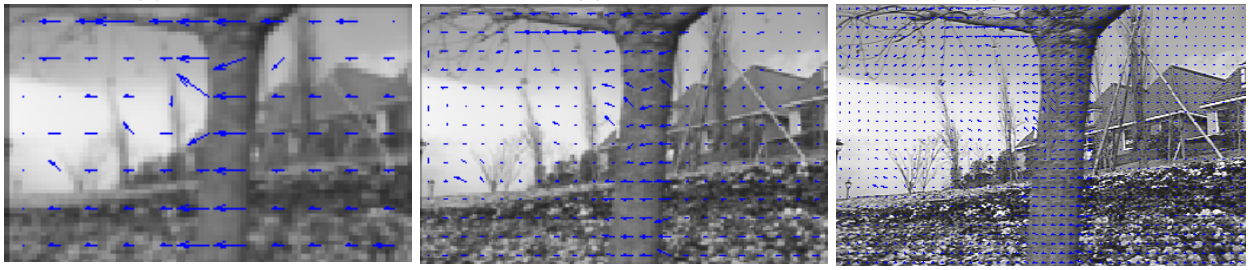
The best results are obtained with the brute force algorithm, but to obtain them it is necessary to use more computational cost and spend more time. So, depending on the problem is best to use one algorithm or the other.

1.4 Multiresolution algorithm

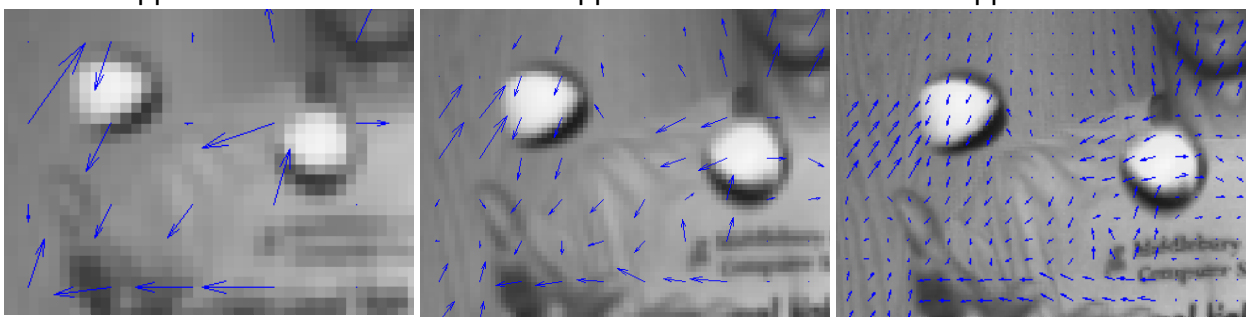
Question: Compare the resulting motion vectors with the ones obtained in section 1.1 and the ones obtained in section 1.3. What do you think are the advantages and disadvantages of the multiresolution scheme with respect the other methods you have seen here ? What about the execution time ? Comment your response.

The multiresolution algorithm uses the full search algorithm for the first iteration with a reduced image. It makes to select the general direction of the entire picture in a non expensive computation time. Once this general vectors are computed, these are used to do a conjugate search with the given direction. It makes to find the local minimum in a direction lined with the rest. This is made with half resolution and finally with the real resolution.
This method seems a good approach because is based in the assumption that the objects in the image will move in the same direction that the low resolution image. It can be a good assumption in some cases, but may be in an other cases the direction of little objects is not the same that the greatest.
It implies that in the image with the tree where we know that all the objects are moving to the left side the Multiresolution approach makes sense. In the other side in the full search block of the first section you need to define which resolution or block size you want. This makes that in a high resolution like block size 4, there are little objects that moves in random directions because it seems to be the same little objects.
The best point in the Multiresolution approach is that even you need to select the block size (like the other algorithms), the different resolutions in some form makes the different block size, and then with a block size of 8 is simulating a 32, 16 and 8 block size (for the 1:4, 1:2, 1:1 resolution respectively).

first approach with 1:4 | second approach with 1:2 | final approach with 1:1



first approach with 1:4 | second approach with 1:2 | final approach with 1:1



About the execution time the fastest algorithm is the Conjugate Search Direction, because it tries to find a local minimum, but only in one defined block size and then big or little objects may not be found. In the contrary, the Full Search Algorithm is very slow finding the best minimum for each block, but like in the local search you need to specify the block size.
And the algorithm with a middle execution time is the Multiresolution, it makes one Full Search but with a 1:4 resolution, that is a very fast search for the low resolution. And at the ending it uses two conjugate searches with the help of the vectors calculated before. For that reason although the resolution would increase the search time, the help of the vectors makes this search fastest.

Method	**time (seconds)**
Full Search (block size = 4)	76.518840
Full Search (block size = 8)	18.922578
Full Search (block size = 16)	4.834862
Conjugate Search Direction (block size = 8)	0.452174
Multiresolution Algorithm (block size = 8) rows below	**1.49336**
- Full Search (resolution = 1:4, block size = 8)	1.029206
- Conjugate Search Direction (resolution = 1:2, block size = 8)	0.114247
- Conjugate Search Direction (resolution = 1:1, block size = 8)	0.349907

2 Non-parametric motion estimation

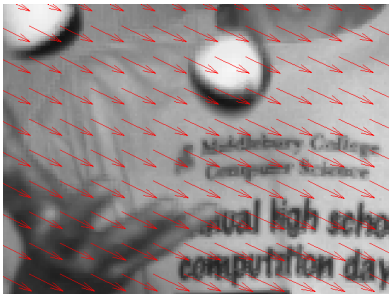
Question: comment on the effect of the value of λ on the estimated flow.

Lambda value seems to decide the velocity of the blocks (the velocity of one block is represented by the distance that this block crosses from one picture to the other). In the next images is possible to see that in the case of the balls, with a lambda value of forty, it shows clearly the direction of the fastest objects like the balls or the hands of the man. In the second case with a lamba value of one hundred thousand it shows the movement of the men instead of the balls, this is not than fast as the balls. In last case with lambda equal to ten million it shows the direction of all the picture, it is produced for the movement of the camera and is very difficult to appreciate in the two pictures.

lambda = 40

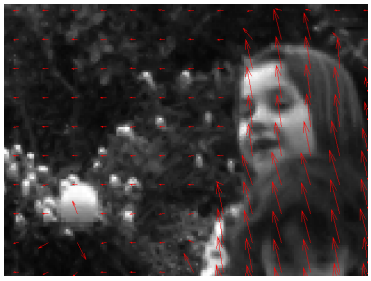
lambda = 10^5

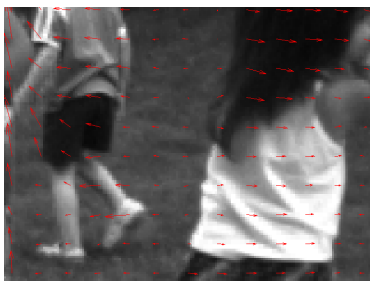
lambda = 10^7



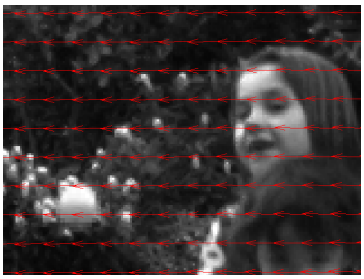
In the case of the backyard pictures, the fastests objects (or blocks) are the ones that represents the two girls jumping in the middle of the image. The second fastest blocks are the little girl at the right of the picture and one little boy. Finally the last value of lambda shows the direction of all the image, representing the movement of the camera.

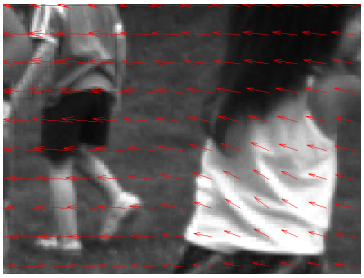
Lambda = 40



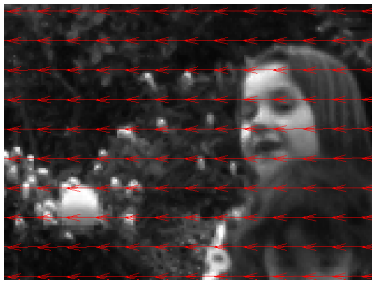


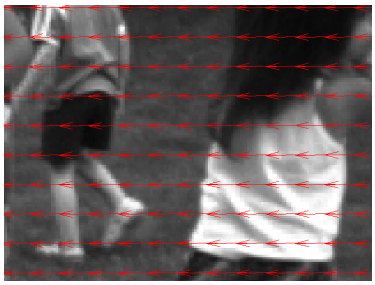
Lambda = 10^5





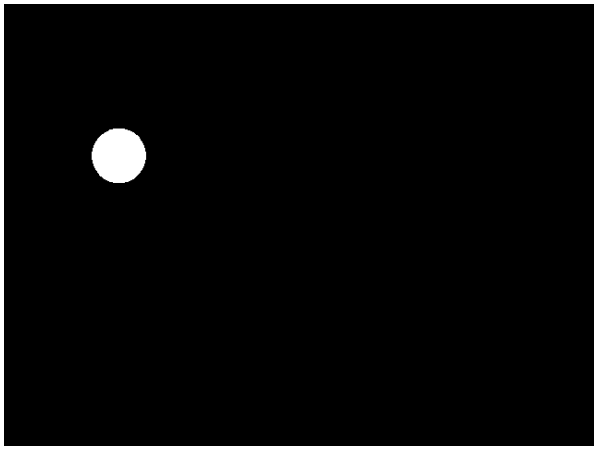
Lambda = 10^7

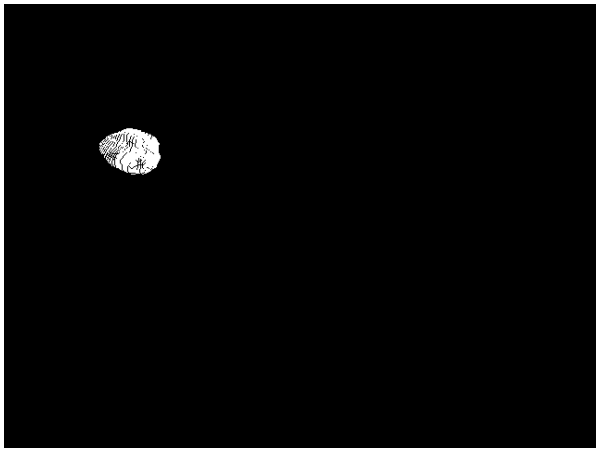


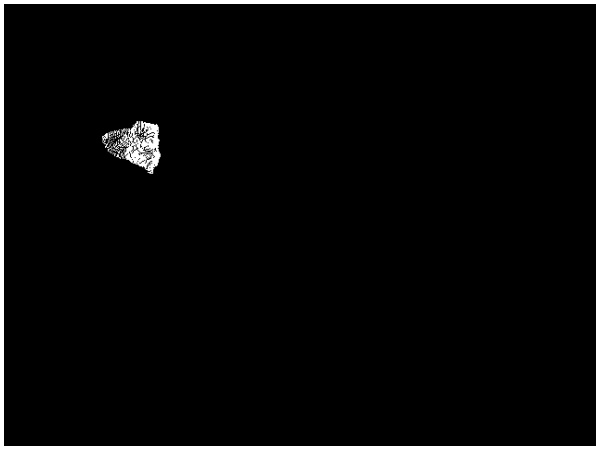


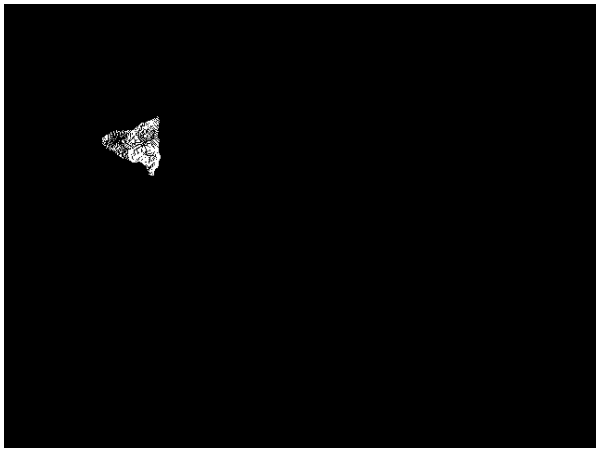
3 A simple tracking algorithm

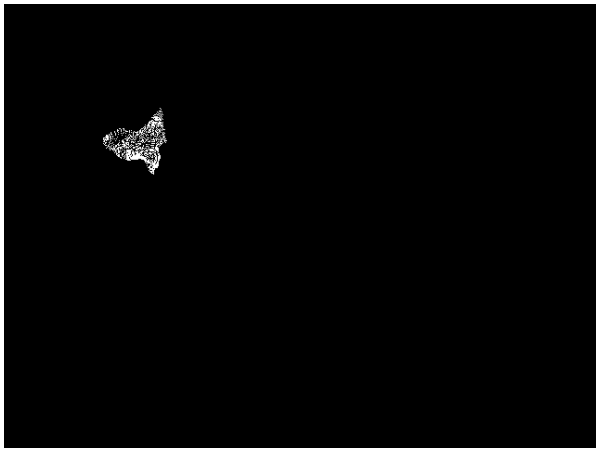
Question: Comment on your experiences with the tracking algorithm and attach the MATLAB code with your deliverable.

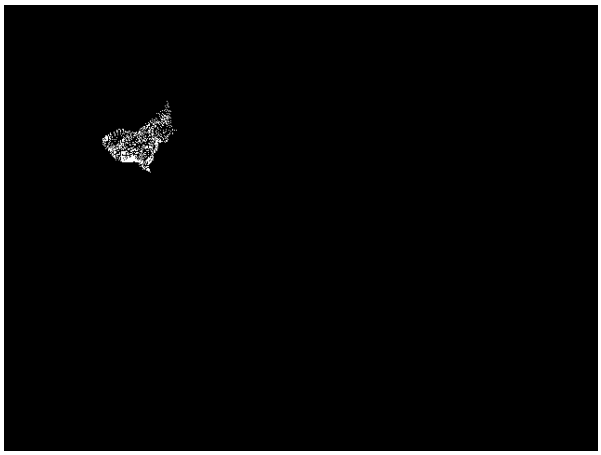



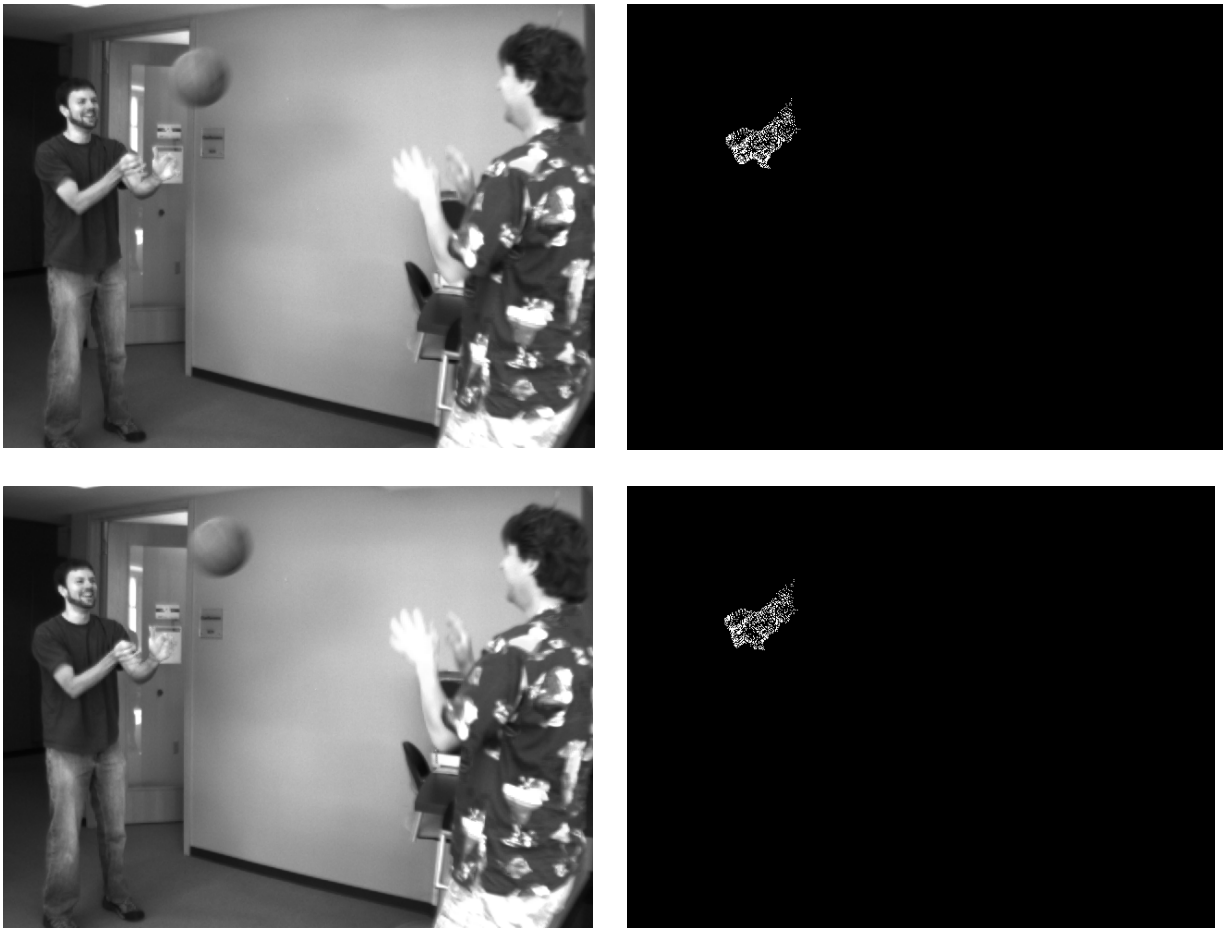










We have performed a non-parametric motion estimation algorithm. As we can see in previous images, the algorithm is able to detect the direction of the motion, and follow the ball, but not very accurately.

As we propagate the mask pixel by pixel, we can see how many of the pixels are correctly calculated, but it is an error of some pixels of the mask which fall outside the ball. These errors are propagated in each sequence, because when a pixel in the mask falls outside the ball, and into the background, in next iteration we are actually performing motion estimation of the background in that pixel. And as the background is not likely to move, the mask pixel remains mostly in the sample place for the whole sequence.

A way to avoid this, would be to average the motion of all pixels, and avoid the errors this way. The drawback is that while non-parametric motion algorithms are able to handle deformation of the moving objects, with this solution we would always have a mask with the same exact shape.