# Stereo vision

A brief introduction

Máté István

MSc Informatics

# What the stereo vision aims

- Retrieving 3D information, and structure of an object with two, or one moving camera. In this project we use one moving camera.
- A line and a plane, not including it, intersect in just one point. Lines of sight are easy to compute, and so its easy to tell where any image point projects on to any known plane.
- If two images from different viewpoints can be placed in correspondence, the intersection of the lines of sight from two matching image points determines a point in 3D space.

# Stereo vision and triangulation

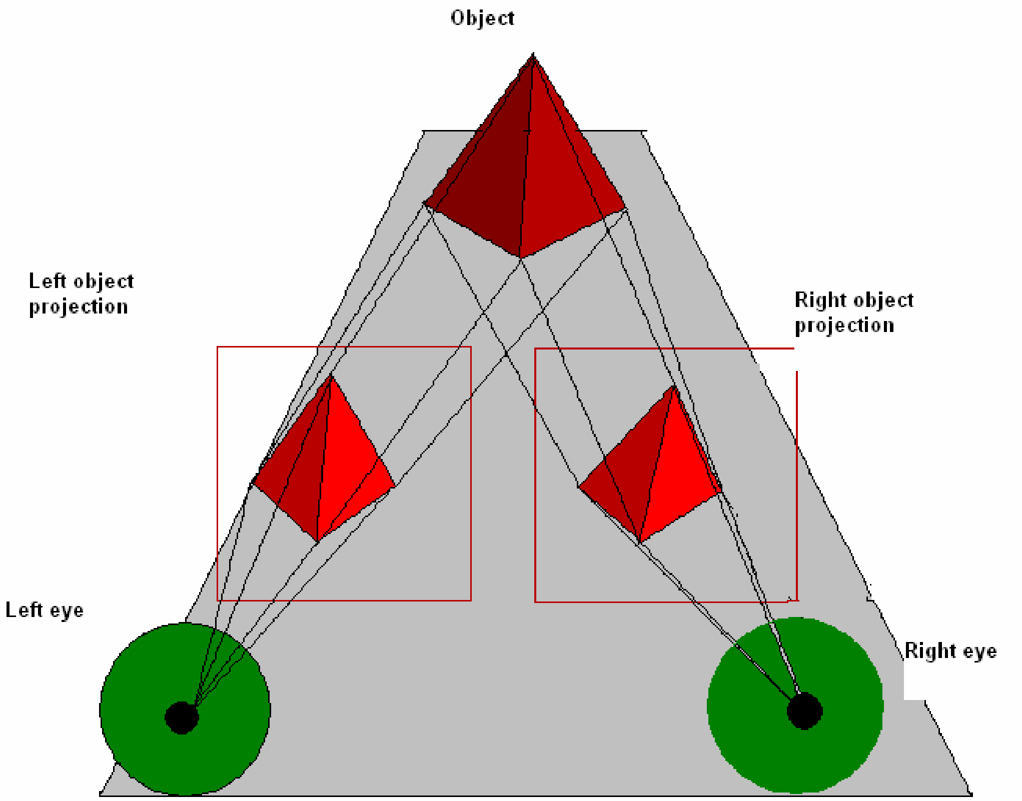
- One of the first ideas that occurs to one who wants to do three-dimensional sensing is the biologically motivated one of stereo vision. Two cameras, or one from two positions, can give relative depth, or absolute three-dimensional location.
- There has been considerable effort in this direction [Moravec 1977, Quam and Hannah 1974, Binford 1971, Turner 1974, Shapira 1974]

# The technique

1. Take two images separated by a baseline
2. Identify points between the two images
3. Use the inverse perspective transform, or simple triangulation to derive the two lines on witch the world points lie.
4. Intersect the lines

The resulting point is in three-dimensional world coordinates.

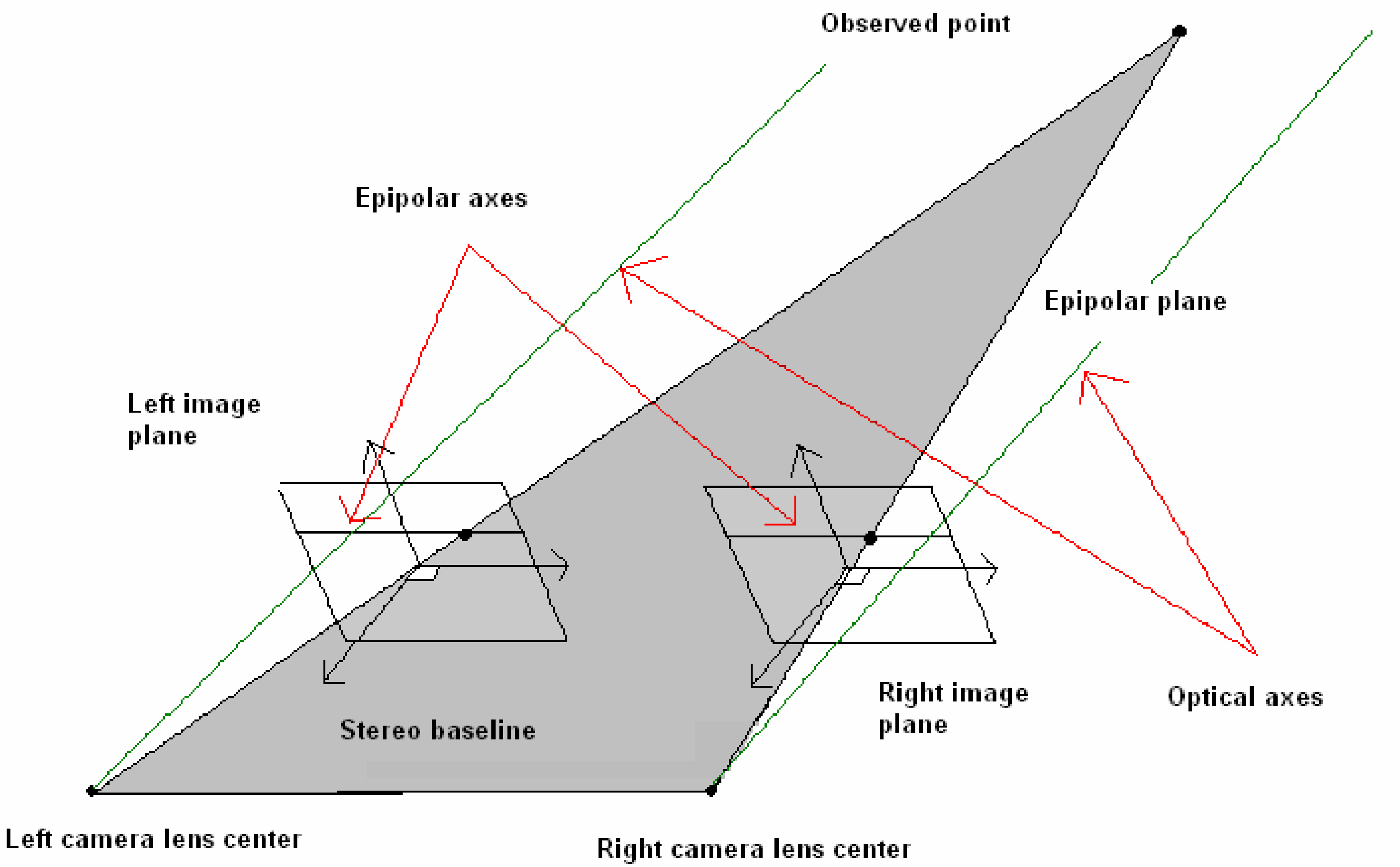
The hardest part of this is method is step 2, that of identifying corresponding points in the two images.


Object
Left object projection
Right object projection
Left eye
Right eye

# Stereo vision terminology

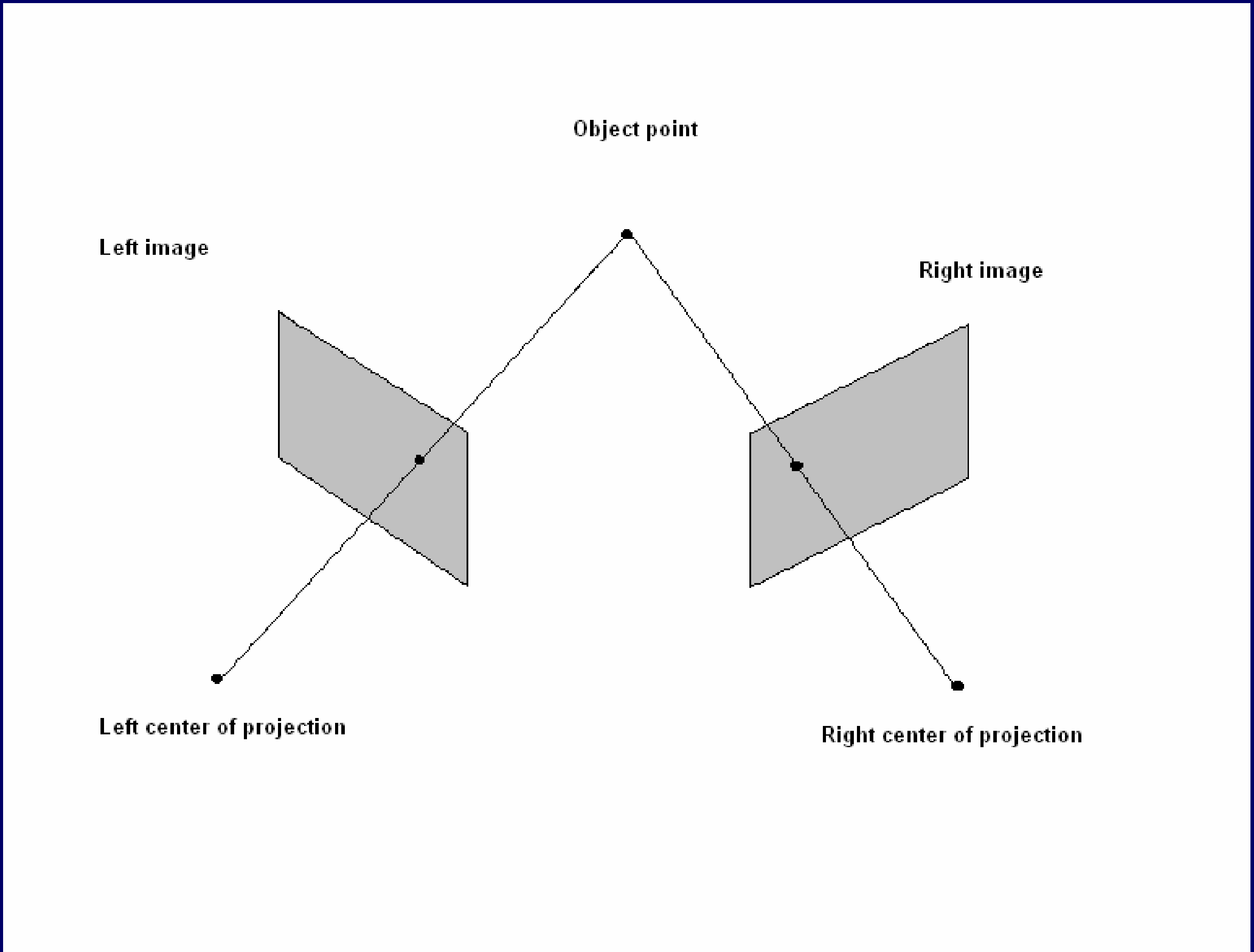
- **Fixation point**: the intersection of optical axis
- **Baseline**: the distance between the centers of the projection
- **Epipolar plane**: the plane passing through the conters of projection and the point in the scene
- **Epipolar line**: the intersection of the epipolar plane with the image plane
- **Conjugate pair**: any point in the scene that is visible in both cameras will be projected to a pair of image points in the two images

- **Disparity**: the distance between corresponding points when the two images are superimposed
- **Disparity map**: the disparities of all points from the disparity map (can be displayed as an image)


Observed point
Epipolar axes
Epipolar plane
Left image plane
Right image plane
Optical axes
Stereo baseline
Left camera lens center
Right camera lens center

# Triangulation-the principle underlying stereo vision

- The 3D location of any visible object point in space is restricted to the straight line that passes trough the center of projection and projection of the object point
- Binocular stereo vision determines the position of a point in space by finding the intersection of the two lines passing through the center of projection an the projection of the point in each image


Object point
Left image
Right image
Left center of projection
Right center of projection

# Two main problems of stereo vision

I. The correspondence problem

II. The reconstruction problem

# I. The correspondence problem

Finding pairs of matched points such, that each point in the pair is the projection of the same 3D point

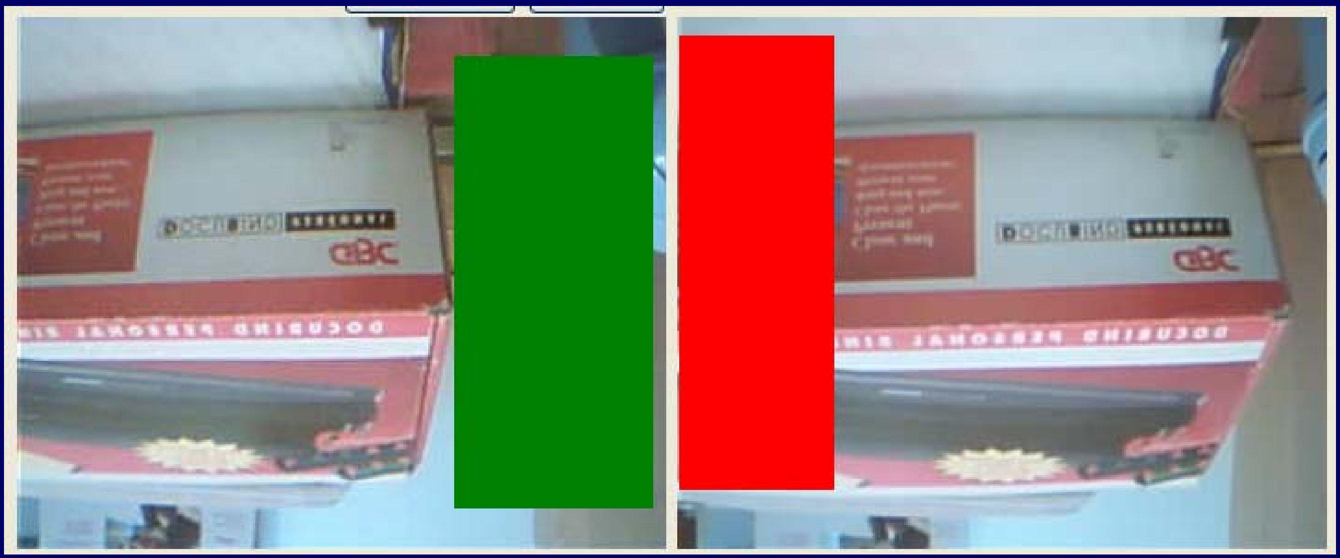
- Triangulation depends crucially on the solution of the correspondence problem.
- Ambiguous correspondence between points in the two images may lead to several different consistent interpretation of the scene

- Efficient correlation is of technological concern, but even if it were free and instantaneous, it would still be inadequate.
- The basic problems with correlation in stereo imaging relate to the fact that objects can look significantly different from different viewpoints
- It is possible for the two stereo views to be sufficiently different that corresponding areas may not be matched correctly
- Worse, in scenes with much obstruction, very important features of the scene may be present in only one view.

- This problem is alleviated by decreasing the baseline, but the accuracy of depth determination suffers.
- One solution is to identify world features, not image appearance, in the two views, and match those (the nose of a person, the corner of a cube)

# Why is the correspondence problem difficult?

- Some points in each image will have no corresponding point in the other image, bacause:
- The cameras may have different fields of view
- Due to occlusion
- A stereo system must be able to determine the image parts that should not be matched.



- In the above picture, the part with green and red are the parts that show the different viewpoint of the cameras
- The task is to find points, that can be seen for both cameras
- Occlusion is both visible at the right edge of the box

# Methods for establishing correspondence

- There are two issues to be considered:
- How to select candidate matches ?
- How to determine the goodness of a match?
- **A possible class of algorithm**
- Correlation based attempt to establish correspondence by matching image intensities

# Correlation-based methods

- Match image sub-windows between the two images using image correlation
- Scene points must have the same intensity in each image (strictly accurate for perfectly matte surfaces only)

# The algorithm

Two images $I_L$ and $I_R$ are given

- In one of the images ($I_L$) consider a sub-window W, in the other a point $P=(P_x, P_y)$
- The search region in the right image $R(p_L)$ associated with a pixel $p_L$ in the left image
- For each pixel $p_L = (i, j)$ in the left image:
  - For a displacement d=(dx,dy) in $R(p_L)$ find
  - C(d) – a “norm” (Euclidian, Minkowski), correlation between the pixel pairs in images

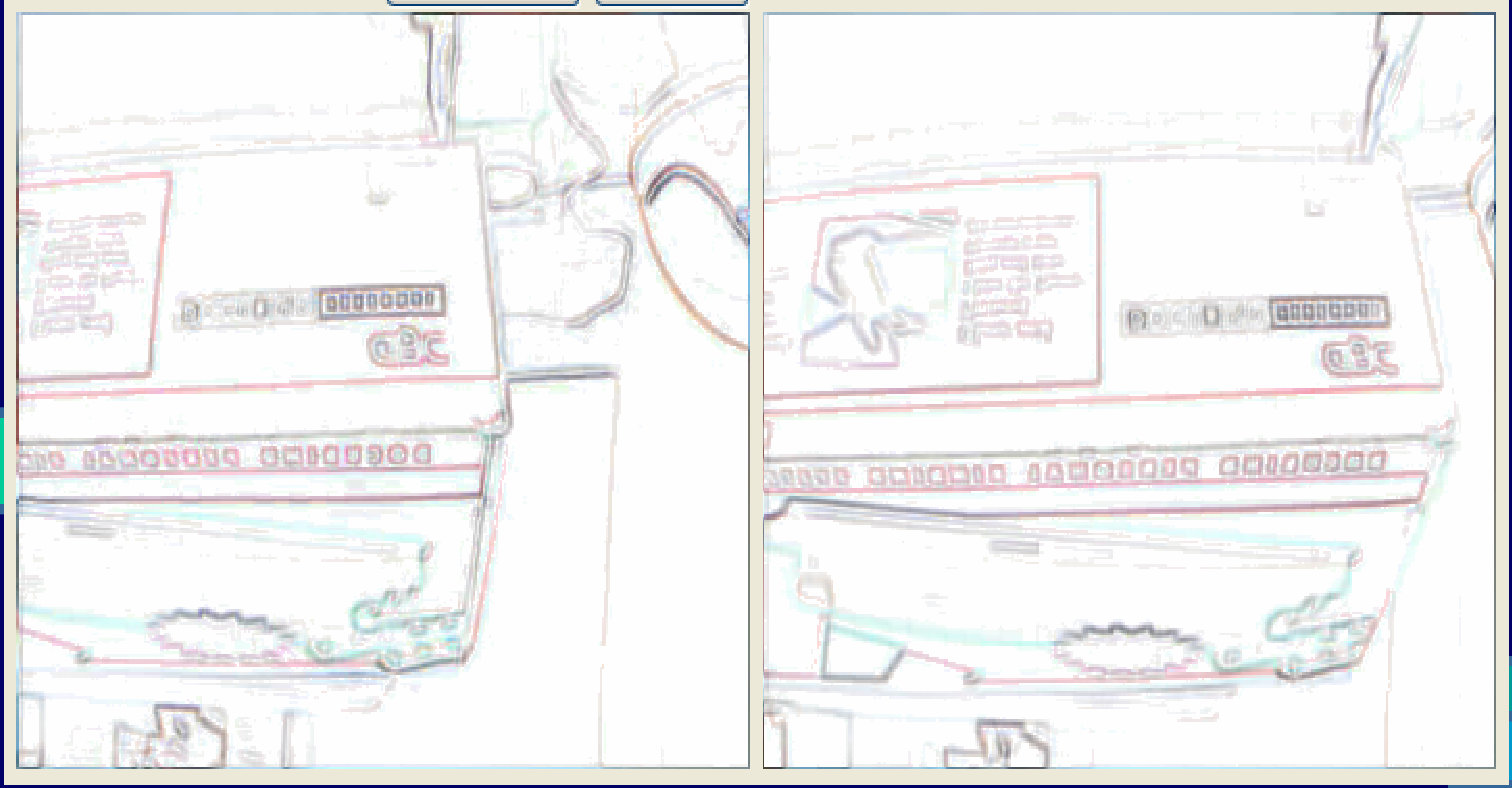
- Example: I choose the absolute difference between RGB pixel values:

  $C(d) = \mathrm{SumAbs}(PV_{(R,G,B)}(W_{ij})-PV_{(R,G,B)}(I_R(P_x+i+dx),I_R(P_y+j+dy)))$

- This expresses that we count the difference
- The disparity of $p_L$ is the vector $d'=(dx',dy')$ that minimizes $C(d)$ over $R(pr)$

  $d' = \arg\min[C(d)]$

- Improvements:
- I used edge-define in each picture, to produce more accurate results, given that I work with colored pictures in RGB space, so the algorithm is more like a feature-based algorithm without rotation and stretching



- The pictures after edge-finding
- This way the matching is more accurate given that, mainly only edges remained

# Problems and “how-to”-s

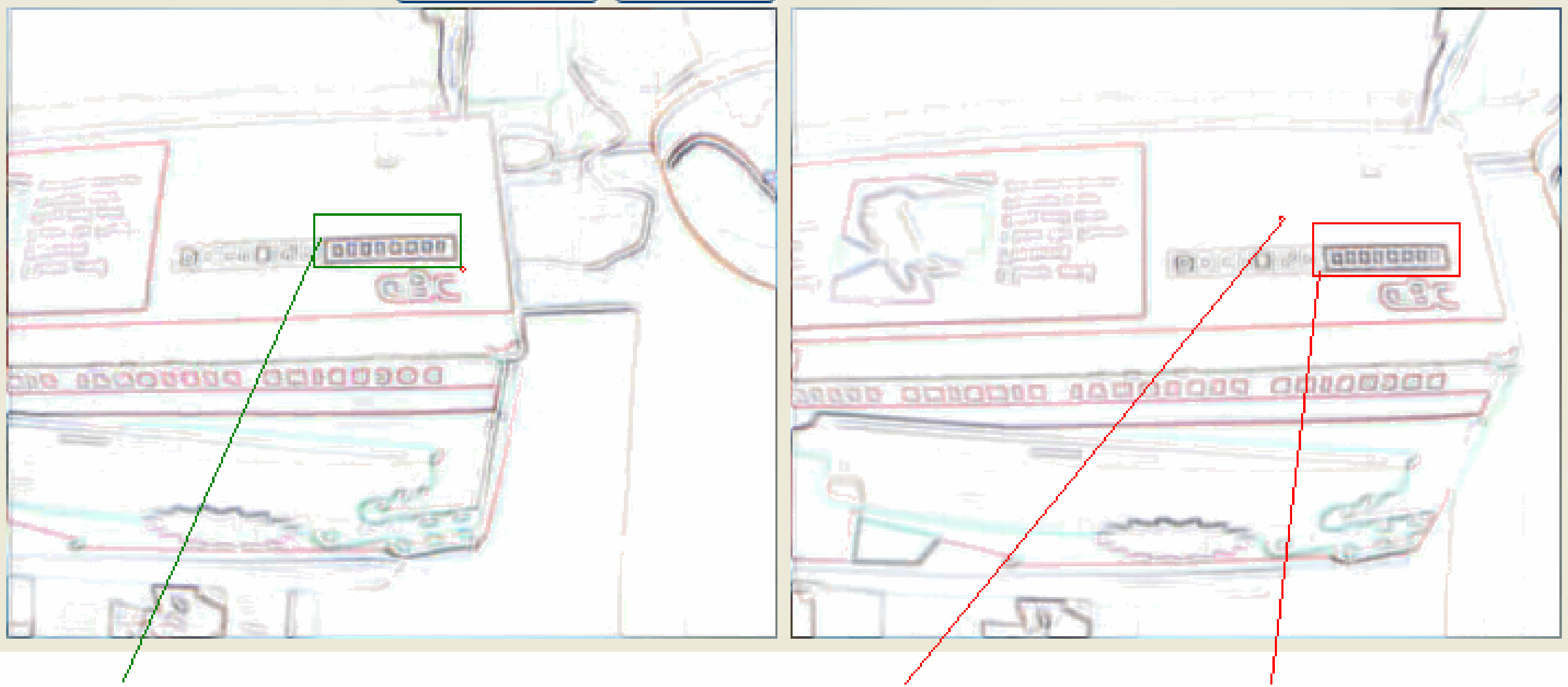
- This algorithm works well in a case of randomly given W sub-window, and a point P, that is that is at a chosen distance d=(Apx,Bpx)
- The question is how to determine the starting d, and the initial W sub-window, knowing that it can be a sub-window **not present** in the other image (due to the different camera viewpoint)

- How to determine a threshold, to speed up computation

- How to determine the sub-window size?

- Too large sub-window becomes inaccurate, due rotation in the images, too small becomes inaccurate due lack of information

- To answer these questions intensive data observation and behavior is needed



Initial subwindow W

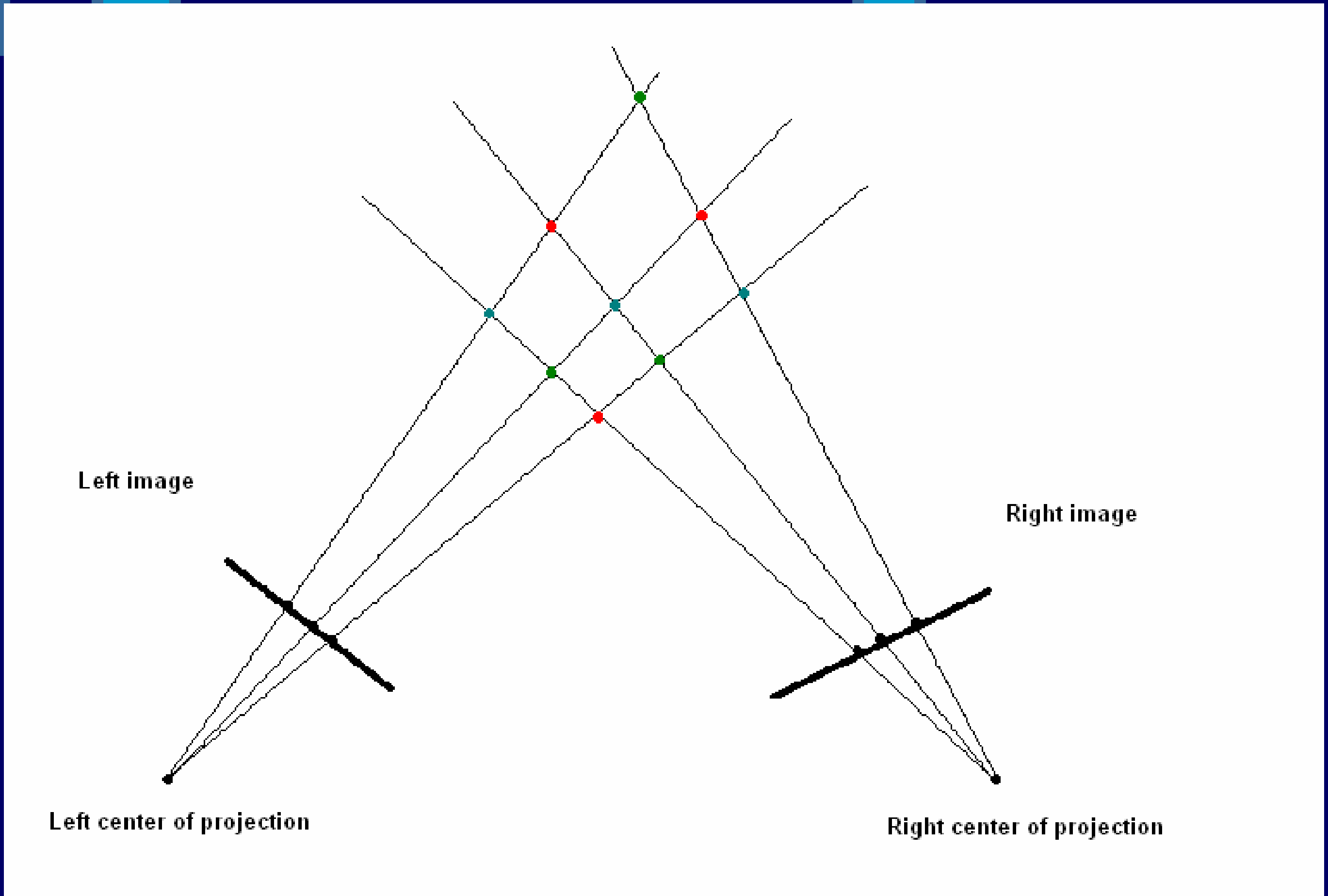
Initial point P

Best matching rectangle
found-with d disparity

- The result for an arbitrary W sub-window and a P point. The result is quite good, but the image rotation can be seen

# The reconstruction problem

- Given the corresponding points, we can compute the disparity map
- The disparity can be converted to a 3D map of the scene



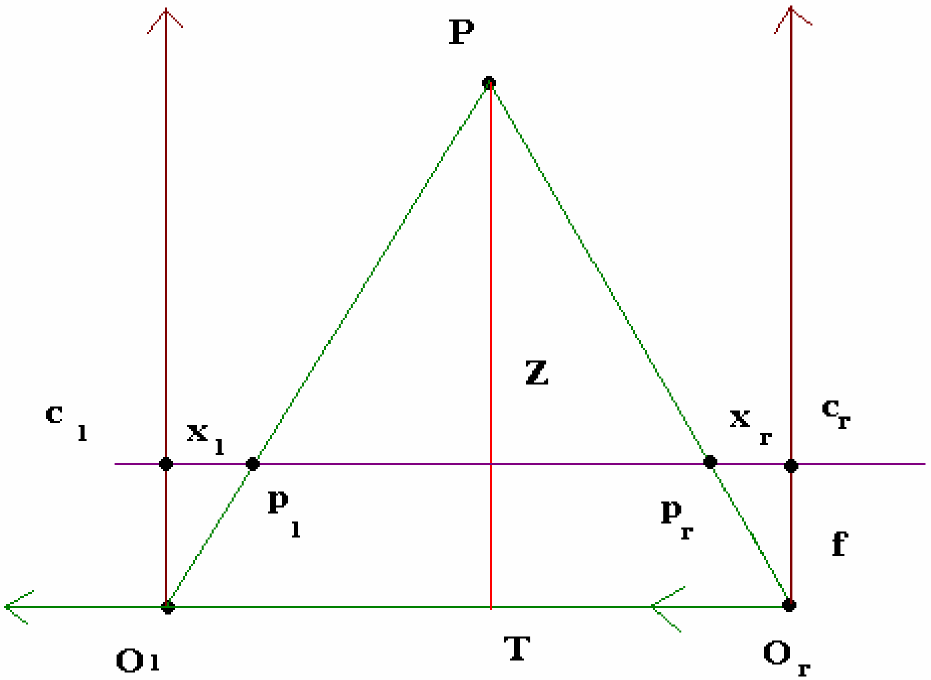
- Incorrect matching can give bad results

# Recovering depth (reconstruction)

- Consider recovering the position of P from its projections $p_r$, and $p_l$ $x_l = f\frac{X_l}{Z_l}, or \quad X_l = \frac{x_l Z_l}{f} \; and \quad x_r = f\frac{X_r}{Z_r}, or \quad X_r = \frac{x_r Z_r}{f}$
- Usually, the two cameras are related by the following transformation $P_r = R(P_l - T)$
- Using $Z_r = Z_l = Z$ and $X_r = X_l - T$, we have

$$\frac{x_l Z}{f} - T = \frac{x_r Z}{f} \quad or \quad Z = \frac{Tf}{d}$$

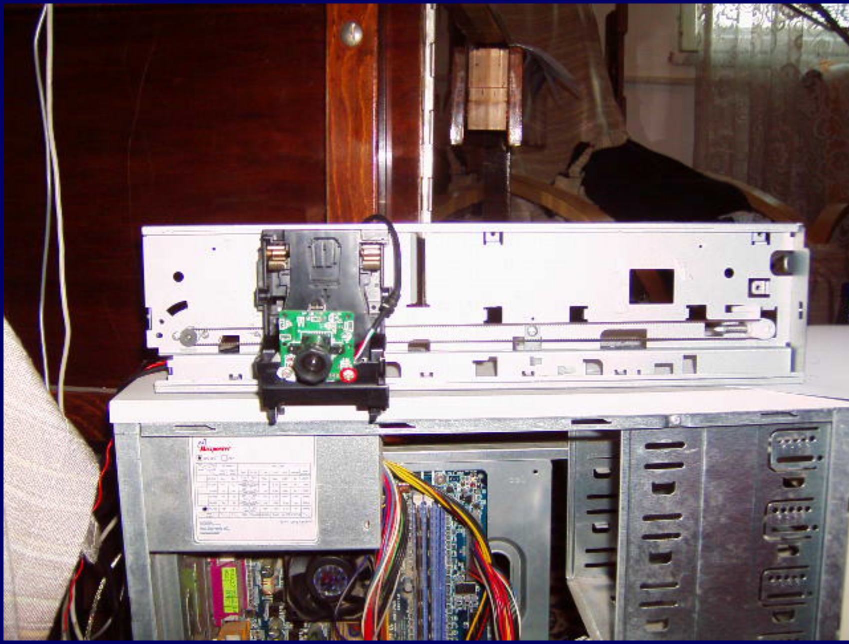
- where $d = x_l - x_r$ is the disparity ( the difference in the position between the corresponding points in the two images )


P
Z
c l
x l
p l
x r
c r
p r
f
O l
T
O r

# The camera

- The camera is set on an old printer, that helps it moving in the same plane.

- This improves the stereo vision with single camera moving in a plane.



# References

[1] Computer vision - Dana Ballard, Christopher Brown

[2] Stereo Camera - *T. Kanade and M. Okutomi*

[3] Why use 3D data ? – Dave Marshall

[4] Methods of 3D acquisition – Dave Marshall

[5] The correspondence problem - *T. Kanade and M. Okutomi*