Local Stereo Matching Algorithm based on guided Markov localization - A report

Manash Pratim Das MPDMANASH@GMAIL.COM
Indian Institute of Technology
Kharagpur, WB, India

1. Introduction

Stereo matching is the most crucial feature required for 3D perception of mobile robot which relies solely on visual information. The idea is to have a robust stereo matching algorithm which has the following features:

- Completely local method: For a disparity of a particular pixel, it should not rely on the complete image, rather process only a local region.
- Guided: Select such local pixels to process which will lead the search to the real disparity. Hence disparity of pixels lying in uniform areas can also be calculated by extending the search towards non-uniform areas of the image.

Such an algorithm will let us compute accurate sparse disparity of the most significant pixels of the image for robot perception. This will effectively reduce computational resources required.

2. Markov Localization Model

2.1 Basic Notation

Let us assume that the stereo images follow epipolar geometry. Let the position of a pixel in y axis of one image be denoted as p and its disparity as d. So, let us denote the position of the corresponding pixel in the other image by $l = p \pm d$. Initially we do not know l and hence we search for it through multiple iterations of "$sense''$ and "$motion''$ methods as in Markov localization of a robot in its environment. Hence, we establish an analogy between robot and the target pixel which are to be localized in its environment.

Let l_i be the location of corresponding pixel after i iterations, and let L_i denote its corresponding random variable. If p is the pixel of left image that we want to match to a pixel on the right image. Then L comprises of the possible positions of that pixel on the right image on the same epipolar line in the range $\{p - d_{max}, p\}$. Initially when l is uncertain we establish an uniform probability distribution over a space of possible positions denoted by $Bel(L_0)$. The belief $Bel(L_i)$ is carried over multiple iterations to make the position more certain.

Similar to the Markov model, we also perform two operations over each iteration. The analogy between the operations is explained in the following:

- Sense: In robot localization, this operation takes sensor readings of its environment and updates the belief $Bel(L_t)$ based on a given map of the environment over time t. Whereas our sense operation reads pixel value p of one image (say left) and attempts to localize its corresponding pixel l on the other image (say right) in L while updating the belief $Bel(L_i)$ over iterations i.

- Motion: In robot localization, this operation moves the robot to some new position (x, y) from where the new sense operation will be performed. Where, we obtain a new pixel p_{new} to compare in L_{new} and to update $Bel(L_i)$. Ideally, the sequence of p_{new} that we select for comparison should be such that it provides unique pattern that can be matched in both the images to localize the l for the p we initially started with.

2.2 Analogy of the algorithms

2.2.1 Markov robot localization

Suppose the world is discrete and given by the color of walls in an arena $W = \{R, G, R, R, G\}$. The robot performs the following set of observation $O = \{G, R, R\}$ and motion $M = \{1, 1, 1\}$ in the world. The sequence comprises of sense (G) and motion (1 towards right) in sequence. Let the uniform probability distribution be P. The sense and motion algorithm performs:

Algorithm 1 Markov Robot Localization

procedure RobotSense(Z, P)
    $q \leftarrow []$
    $pHit \leftarrow 0.6$
    $pMiss \leftarrow 0.2$
    for $i = 0$ to length of P **do**
        $hit \leftarrow (Z == W[i])$
        $q[i] \leftarrow P[i] * (hit * pHit + (1 - hit) * pMiss)$
    $Normalize(q)$
    return q
procedure RobotMove(U, P)
    $q \leftarrow []$
    for $i = 0$ to length of P **do**
        $s = P[(i - U)\%$ length of $P]$
        $q[i] \leftarrow s$
    return q
procedure Localize
    $P \leftarrow []$
    for $i = 0$ to length of O **do**
        $P = RobotSense(O[i], P)$
        $P = RobotMove(M[i], P)$

The $argmax(P)$ after all the operations Z and U in O and M respectively on W will give the current location of the robot which can be used to find the initial position. The above operation will output: $P = [0.066, 0.066, 0.2, 0.066, \mathbf{0.6}]$

2.2.2 Stereo Matching

Since the pixels are already discrete hence the world can be considered to be the gray-scale pixel values $W = \{1, 10, 1, 1, 10\}$. This world as described above is generated from the pixel values of random variable L. For example, we need to match a pixel from left image with p (intensity given by $I_l(p)$) as its position on x axis to its corresponding pixel on the right image (intensity given by $I_r(l)$). Then we can define L as $L = \{p - d_{max}, p\}$ as the required pixel will be found towards left due to shift between the two images and in the same epipolar line.

We now need a measure to define similarity between pixel values for comparison. In this algorithm the binary parameter hit is defined as: $(abs(Z - W[i]) < similarity\_threshold)$ where $(similarity\_threshold = 3)$ provided best results in our trials.

W and O belongs to two different images of a stereo pair as we are matching one pixel from the observation of its nearby pixel values O to W with similar pattern. The modified sense and motion algorithm are as follows:

Algorithm 2 Stereo Matching

procedure StereoSense(P)
    $q \leftarrow []$
    $pHit \leftarrow 0.6$
    $pMiss \leftarrow 0.2$
    $similarity\_threshold \leftarrow 3$
    $d_{max} \leftarrow 25$
    for $i = -d_{max}$ to 0 **do**
        $diff \leftarrow abs(I_l(p + motion\_index) - I_r(p + motion\_index + i))$
        $hit \leftarrow (diff < similarity\_threshold)$
        $q[i] \leftarrow P[i] * (hit * pHit + (1 - hit) * pMiss)$
    $Normalize(q)$
    return q
procedure IndexMove(U)
    $motion\_index \leftarrow motion\_index + U$
procedure GetCorrespondingPixel(p)
    $motion\_increment \leftarrow 2$
    $num\_motion \leftarrow 5$
    $P \leftarrow []$
    for $i = 0$ to num_motion **do**
        $P = StereoSense(P)$
        $P = IndexMove(i * motion\_increment)$
    return $argmax(p - d_{max} + P)$

Disparity d is given by $abs(p - GetCorrespondingPixel(p))$. The values $motion\_increment$ and num_motion are to be optimized for better accuracy and will be discussed below. d_{max} can be set by any method that can estimate the maximum disparity that might be possible. One such way could be $d_{max} = width\_of\_image * 0.7$. Suppose we need to match p in $L = \{p - d_{max}, p\}$, then compare $I_l(p)$ with all $I_r(l)$. $motion\_index$ is used to shift index of the base pixel used for comparison in a new set L shifted equally just like the motion model in Markov Localization.

3. Results

The motion (change in $motion\_index$) we take on the image is crucial. The results of two motion models are displayed below.

Figure 1: Test Stereo Image Pair

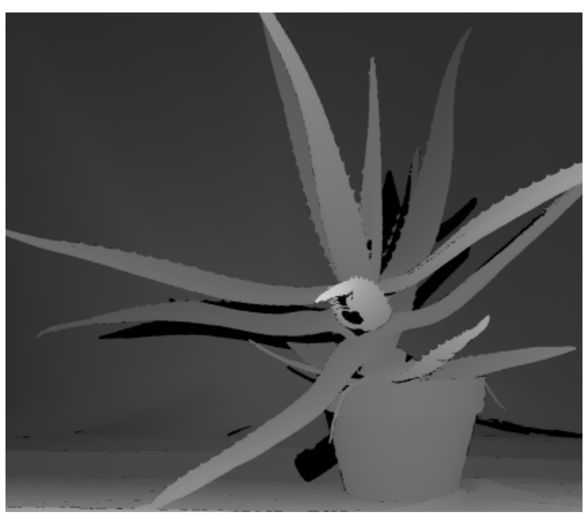

Figure 2: Disparity Ground Truth

3.1 Moving along the epipolar line

$motion\_increment = 1$, $num\_motion = 10$ and hence $motion\_index = \{0, 1, 2, ..., 9\}$

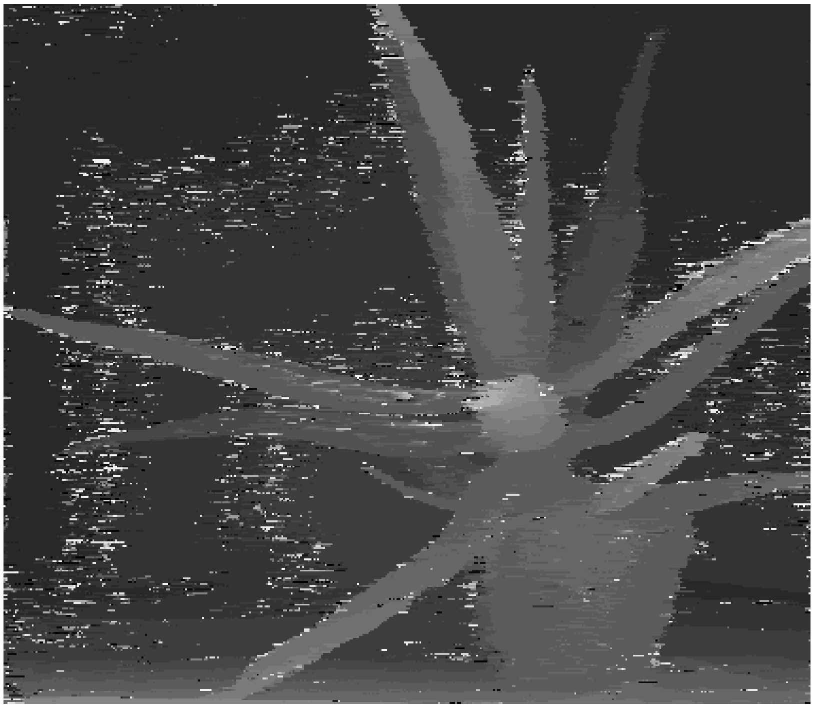

Figure 3: Resulting disparity from linear motion

3.2 Moving in spiral outward motion

In this method the motion is done on both x and y axes in a spiral outward fashion as shown below:

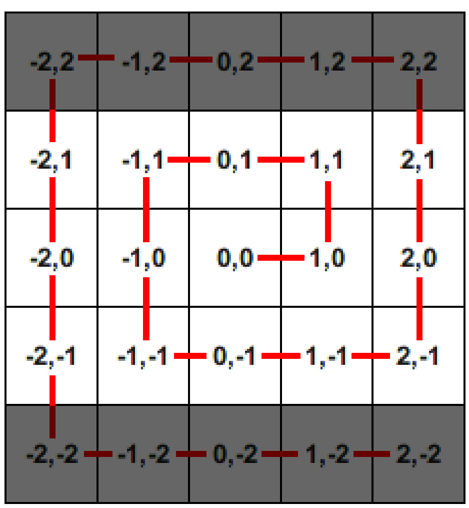

Figure 4: Spiral outward motion starting from 0,0

We can see that, the change in motion model has significant effect in the resulting disparity as shown below.

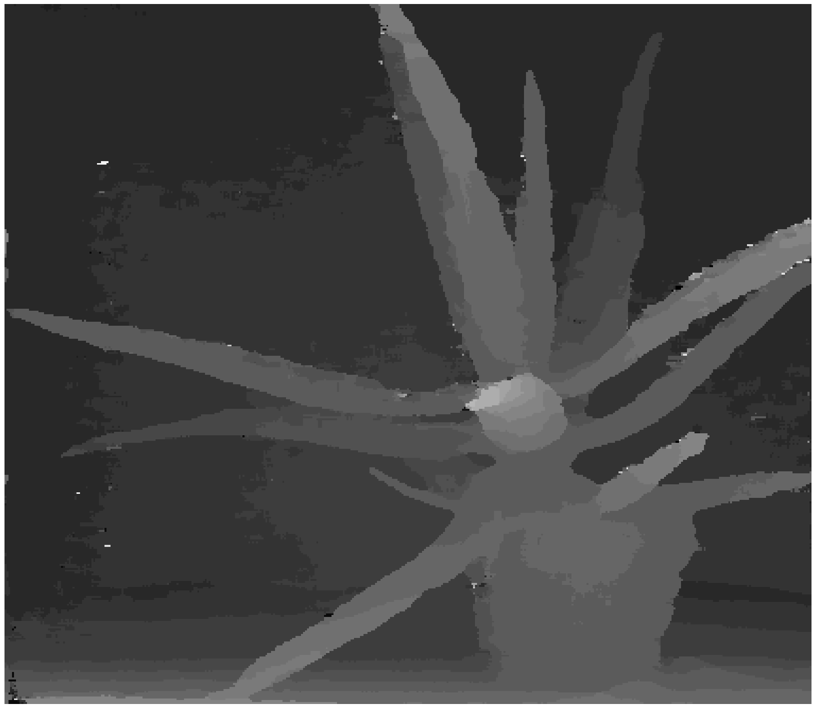

Figure 5: Resulting disparity from spiral outward motion

These disparity maps are the output of the proposed local matching algorithm without any further global processing like sub-pixel estimation and smooth filling.

4. Further development

The ultimate goal is to develop the best motion model which can lead the search towards higher accuracy. The following ideas are yet to be tested:

- Use color information to guide the search such that around 75% of the probability distribution comprises of the same object as the target pixel and rest 25% lies in boundary of the object. The similarity of the pixels based on color can be measured by truncated absolute differences of color. This model is borrowed from the optical flow literature and is given as:

$$M(p,d) = \sum_{i=1}^{i=3} \mid I^i_{left}(p) - I^i_{left}(p-d) \mid \tag{1}$$

$$G(p,d) = \mid \Delta_x(I_{left}(p)) - \Delta_x(I_{right}(p-d)) \mid \tag{2}$$

$$C(p,d) = \alpha * min(T_c, M(p,d)) + (1-\alpha) * min(T_g, G(p,d)). \tag{3}$$

Here, $I^i(p)$ denotes the value of the i^{th} color channel in RGB at pixel p, $\Delta_x(I(p))$ denotes the gradient in x direction computed at pixel p, α balances the color and gradient terms and T_c, T_g are color and gradient truncation values.

- Use edge information to guide the search mostly inside the same edge boundary to better resolve small objects and reduce edge blurring.
- Submit for Middlebury Stereo Evaluation.

5. Links and References

1 http://www.cs.cmu.edu/afs/cs/project/jair/pub/volume11/fox99a-html/node2.html

2 http://vision.deis.unibo.it/ smatt/Seminars/StereoVision.pdf

3 https://publik.tuwien.ac.at/files/PubDat_206200.pdf

Updated on: 6 December 2016