VARIABLE RESOLUTION VISION SYSTEM IN MOBILE ROBOTICS

Armando Sousa[1], Paulo Costa[2], António Moreira[2]

Faculdade de Engenharia da Universidade do Porto (FEUP)
Instituto de Sistemas e Robótica
R. Dr. Roberto Frias, S/N / 4200-465 Porto / Portugal
[1] Lecturer and PhD. Student at FEUP
[2] Professor at FEUP
asousa@fe.up.pt,paco@fe.up.pt,amoreira@fe.up.pt

Abstract: Onboard cameras are the main sensor in many mobile robots applications. High quality cameras require significant computing power to deal with large images. Real Time constraints further emphasize the need for fast and predictable image processing. Taking advantage of some known camera orientation parameters it is possible to reduce the number of interesting pixels by using variable resolution over the image. The actual resolution can be set as a function of presumed feature density. Some results from practical applications are shown and discussed.

Keywords: Autonomous mobile robots, Robot Vision, Robot Navigation, Robotics, Computer Vision, Real-Time

1. INTRODUCTION

High quality cameras require significant computing power to deal with the resulting large images. Using image-processing techniques it should be possible to quickly retrieve the interesting information from the image and present it concisely. Good image processing is essential to Real Time compliance.
This paper presents strategies for dealing with large amounts of information based on assumptions suitable for mobile robotics and other vision problems. The basic concept is to sub-sample the image at a frequency compatible to the predicted object density in that region. Implementation details are taken from the 5dpo-2000 Robotic Soccer (RoboCup) team.

2 MOBILE ROBOTIC VISION PROBLEM

Vision systems frequently have cameras with a certain orientation viewing a scene not perpendicular to the optical axis. This is true for both onboard cameras of mobile robotics as well as for a camera viewing robots tilted. Typical mobile robotics applications exemplified in RoboCup are shown in Fig.1(a) and Fig.1(b); a generic scene is also depicted in Fig.1(c).

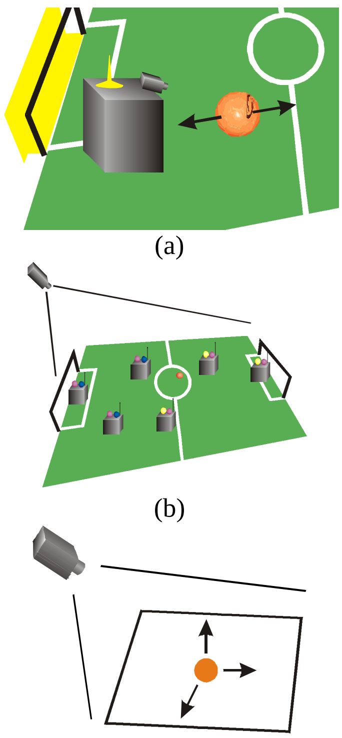


Fig.1. Illustration of vision problems: (a) Camera Onboard; (b) Static External Camera ; (c) generic object moves in 3D over a plane

As illustrated in the examples of Fig.1 the resulting mapping of the world coordinates of the objects and the corresponding coordinates on the camera are not linear (Moreira *et al* 2001). This is due to the fact that the horizontal floor plane where the ball moves is not parallel to the plane where the image is formed.

2.2 General Camera Model

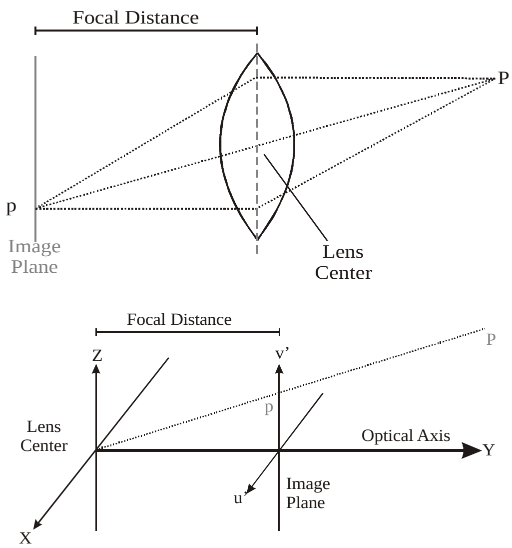


Fig.2. Pin-Hole Camera Model

Assuming the widely known the pinhole model for the camera (Eugene Hecht 1987), such as in Fig.2 it can be seen that there is a projection between the two planes.
The relation between the world coordinates of a point P(X,Y,Z) and the coordinates on the image plane (u',v') in a pinhole camera is shown in (**1**) where f is the focal distance of the lens.

$$\begin{cases} u' = f \cdot X / Y \\ v' = f \cdot Z / Y \end{cases} \qquad \mathbf{(1)}$$

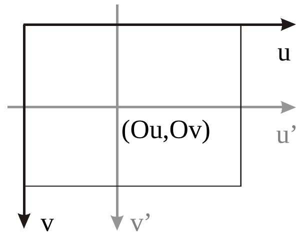


Fig.3. Image coordinates relate to frame coordinates

Image coordinates are related to frame coordinates (Fig.3) as stated in (**2**), where Su, Sv are the horizontal and vertical distances of two adjacent pixels in the frame buffer. If the pixel is square, then Su=Sv.

$$\begin{cases} u' = (u - Ou) \cdot Su \\ v' = (v - Ov) \cdot Sv \end{cases} \qquad \mathbf{(2)}$$

Lens distortion (Eugene Hecht 1987) can be modeled by (**3**) where (xd, yd) are the image coordinates of the distorted image, $rd = \sqrt{(xd^2 + yd^2)}$ and k_1 is a constant depending on the distortion of the lens:
The relation between image and frame coordinates in presence of lens distortion is presented in (**3**).

$$\begin{cases} u' = (\,ud - Ox\,) \cdot Su \cdot (1 + k_1 \cdot rd^2\,) \\ v' = (\,vd - Oy\,) \cdot Sv \cdot (1 + k_1 \cdot rd^2\,) \end{cases} \qquad \mathbf{(3)}$$

Using formula (**3**), it is possible to correct lens distortion as shown in Fig. 4.

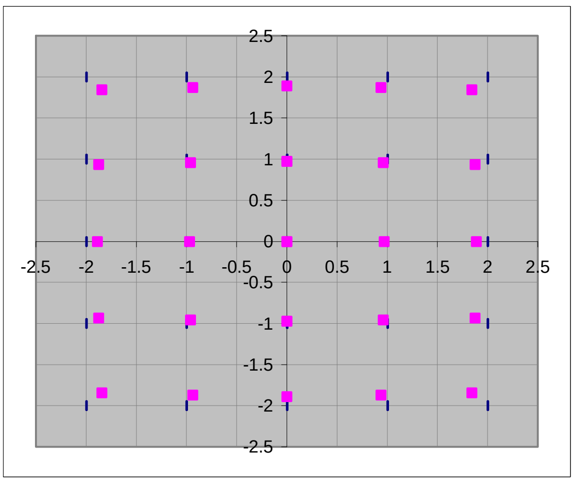


Fig. 4. Correction of lens distortion by means of camera calibration

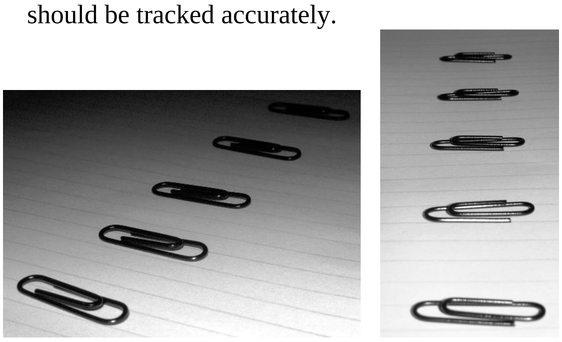
Another issue to address is the camera resolution. If the distances of two adjacent cells in the camera digitalization device are known (Px, Py), then the distances of pixels in the image are given by (**4**) where ax is a scale factor due to the displacement in horizontal scanning and sampling frequencies.

$$\begin{cases} Sx = ax \cdot Px \\ Sy = Py \end{cases} \qquad (4)$$

3 Proposed Algorithm

Common world knowledge and the relations derived earlier allow the following conclusions:

1. Objects closer have larger images (more pixels associated with the same object)
2. Position changes for object closer cause greater change in the image that is to say object far away should be tracked accurately.



(a) (b)

Fig.5. Examples of common projection effects

Although not constant, there is a known relation between image position and real world position for objects placed on the ground.
If objects are closer it is probably a waste of computing power to sample every adjacent pixel of the same object. This is especially true if minimum object size is known. If objects were further away then as much resolution as possible would be interesting. It seems logic to introduce variable decimation of an image having higher resolution to accommodate predicted distance. The same object seen at different distances would not have such different pixel counts and objects moving far away will be detected more accurately.
The new resolution should then reflect predictable object size over the image. For example in the vision system of Fig.5 (b) the objects on the bottom of the image are always bigger than objects on the near the upper side, even if the scene changes. The image should then be sampled taking in consideration a density grid produced with known relations between world and image planes.

3.1 Density Grid Generation

The density grid should be generated by use of the formulas that relate image coordinates and pixels after their "projection" to the real world. Pixels are magnified to different sizes because they are "projected" different distances. In order to keep the same spatial sampling rate, distant parts of the image are re-sampled at higher frequencies.

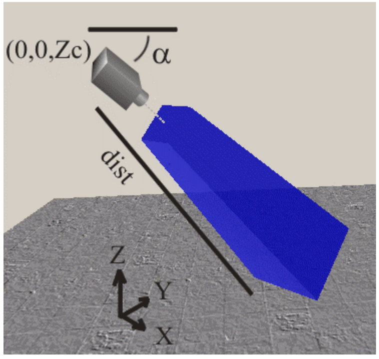


Fig. 6 Diagram for density grid generation showing pixels being projected

$$Distance^2 = \frac{Zc^2\,(\sin(\alpha)\,(Ov - v)\,S + \cos(\alpha)\,f)^2}{(-\cos(\alpha)\,S\,Ov + \cos(\alpha)\,S\,v + \sin(\alpha)\,f)^2} + \frac{Zc^2\,(Ou - u)^2\,S^2}{(-\cos(\alpha)\,S\,Ov + \cos(\alpha)\,S\,v + \sin(\alpha)\,f)^2} + Zc^2 \qquad (5)$$

The generic formula (**5**) was used in to generate pixel's distances. For the sake of simplicity, this formula does not include lens distortion mentioned in equation (**3**) and hence in real applications these formulas must be applied sequentially. As shown in Fig. 6, the position of the camera must be (0,0,Zc), and the camera optical axis must be aligned with the YY axis. The camera is rotated down from the "horizontal" plane XY an angle of α. The formula also assumes square pixels, S=Su=Sv.
As mentioned earlier, the pixel density must be proportional to the distance calculated for each pixel. It now seems convenient to introduce the "horizon" concept, which is a distance in meters that will provide density 1 (full resolution). To prevent densities above 1, formula (**6**) makes use of the minimum operator.

$$\text{Density} = \min(\text{distance}/\text{horiz}, 1) \qquad (6)$$

The densities can now be generated for all pixels and it is possible to plot a density chart along image coordinates u and v, shown in Fig. 7. The following discussion will be generic but the images and plots refer to 5dpo-2000 data.

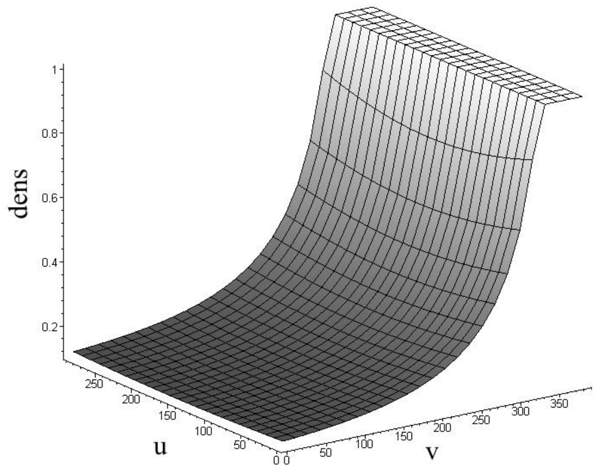


Fig. 7 Plot of expression (**6**), the density formula for the 5dpo-2000 case

Generation of the grayscale densities bitmap is now easy. Such bitmap is shown in Fig 8.

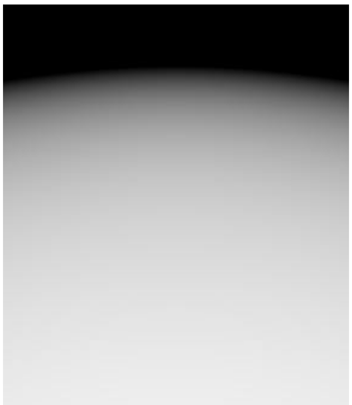


Fig 8 Grayscale densities bitmap for the 5dpo-2000 case. Note camera is portrait position; bottom is nearer.

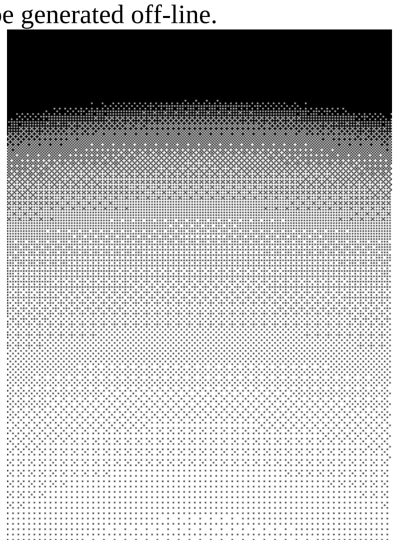
This bitmap depends on internal parameters of the camera like lens distortion, focal distance, viewing angle and CCD resolution. It also depends on camera external parameters like its position and angle. The size of the bitmap is the same as the full image.

After having densities encoded in shades of gray, it is possible to use standard dithering techniques to convert the bitmap to pure black and white. This will generate a bi-dimensional list (bitmap) that marks pixels to analyze or not that comply with the local densities earlier specified.

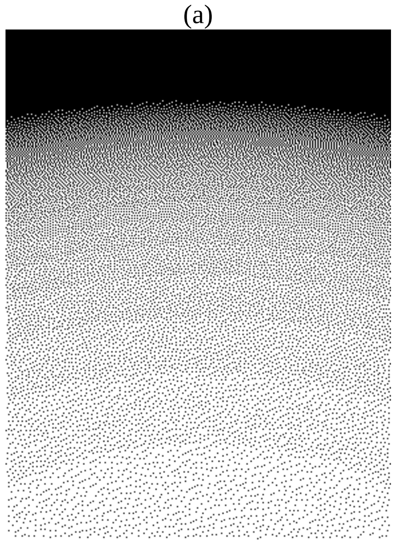
Floyd Steinberg error diffusion (Floyd et al 1976) is a commonly used technique for converting grayscale to pure Black & White images (B&W). This technique is also used for other error spreading algorithms when “fractional” working is necessary and only “integer” positions are available. The fundamental principle is extremely simple: accumulate fractional amounts at each step, but output only the integer portion. Iterate form left to right the error from pixel x to the neighboring pixels as show in (**7**) to obtain a pattern that globally complies with the intended “fractional” working.

$$\begin{array}{ccc} & x & {}^{7}\!/_{16} \\ {}^{3}\!/_{16} & {}^{5}\!/_{16} & {}^{1}\!/_{16} \end{array} \qquad (\mathbf{7})$$

The normal Floyd Steinberg algorithm can be improved as suggested in (Zeggel *et al* 1993) with random noise addition to produce results shown in Fig. 9b where dither image without visible patterns can be seen. The dither pattern is important to prevent correlation with any portion of the image. Correlation effects may lead to a skewed estimate of some of the image's features and thus other methods like ordered dither (Fig. 9a) are less suited to this application. The reason for this is that in the presence of periodic noise it is preferable to have less constant sampling frequencies. Computational power at this point is not an issue for Real Time compliance, as the bitmap should be generated off-line.



(a)



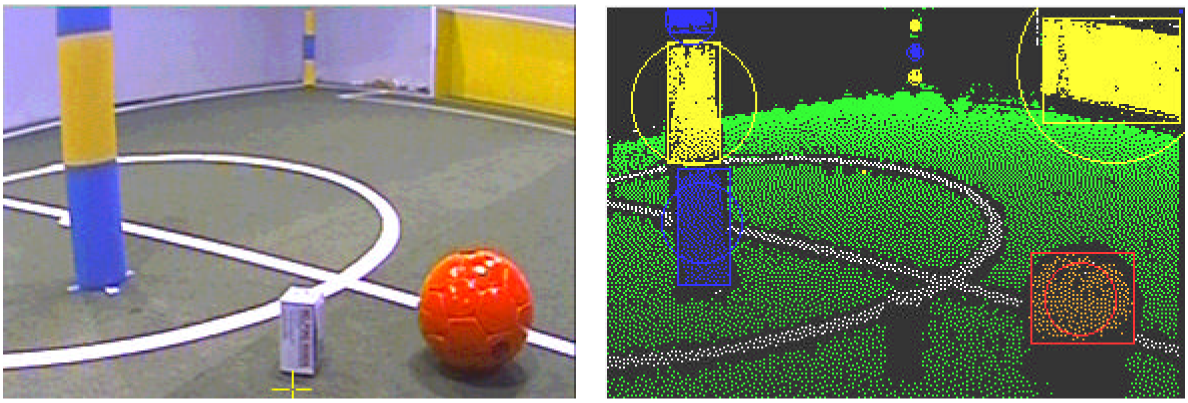
(b)

Fig. 9**.** Bitmap from in Fig 8 converted to pure black and white: (a) Ordered diffusion creates visible patterns (b) Floyd-Steinberg algorithm with added noise does not

An example of a dithered image is shown in Fig.10.

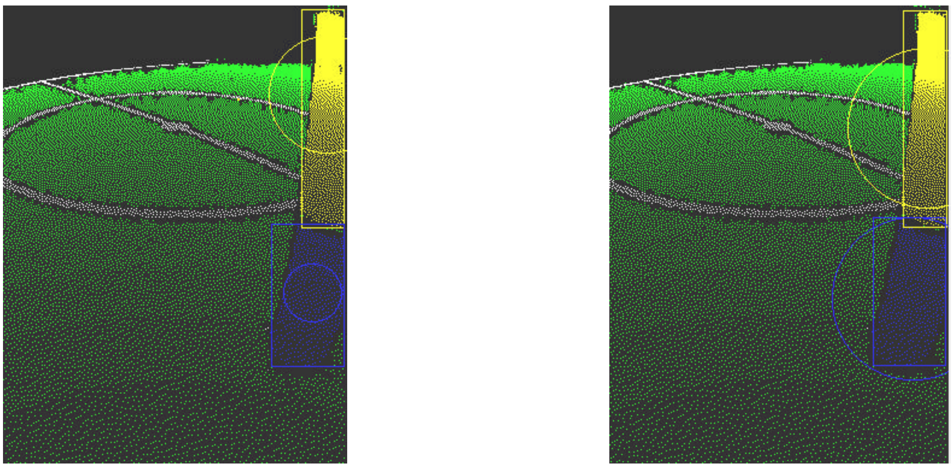
3.2 Pixel weight

The presented model assumes that all objects are near the horizontal plane. The next level of our image processing algorithm uses clusters of pixels to find the objects. If the object is horizontal then the center of the cluster is the center of the object.
Let us now consider an object whose visible face is manly vertical. If all pixels weight the same, the center of the clusters will tend to be distorted to the far sections of the image where there are more pixels. When clustering it is then necessary to consider a weight related to the local density in that area. That information is present in the grayscale bitmap as a numeric information encoded in color. If pixel density is large, the pixel should weigh less, as there will be many pixels in this region. Note in Fig.11b how the centers of the clusters are more accurately calculated. This effect is important when viewing tall objects.



(a) (b)

Fig.10.Example of the method: (a) Real image and (b) the dithered image of the 5dpo-2000 Test Field



(a) (b)

Fig.11. (a) Error in center of cluster due to Center of mass calculations
(b) taking local densities into consideration

3.3 Full Algorithm Reviewed

The overall new algorithm thus includes:

1. **Generate Densities Bitmap:** Using the formulae of predicted distances, generate pixel densities, encode them in a grayscale bitmap and finally convert and dither the bitmap to B&W. This is the Densities Bitmap and is generated off line.
2. **Pixel Classification:** for the black pixels in the Densities Bitmap, classify its color according to previous calibrations. Run the pixel color through the calibration table.
3. **Clustering:** Cluster data structure is: color, number of pixels, center and rectangular limits. If a meaningful color was found during pixel classification, then search all the clusters of that color. Find one cluster that will not expand more than a given threshold and update cluster data structures.
4. **Merging:** If the pixel distance to two clusters is below another threshold, then merge these clusters together and update data structures. Center of the cluster is found by center of mass calculations.
5. **Iterate Image:** Iterate for the whole image to take advantage of cache optimizations of recent computer architectures.

4 Technological Issues

The 5dpo-2000 image-processing was based on a conventional color-clustering algorithm. Additional features included virtual sensors (dominant colors in regions) and edge detection along interesting lines.

The previous algorithm was based on a resolution of 192x144 at 15 bits of color depth. The cameras used are Philips surveillance cameras and the frame grabbers are based on the common BT 878 video acquisition board. This kind of boards allows the frame grabber to access the memory through DMA and so image transfer is done without spending CPU time. The whole image processing took as little as 7 ms in a Celeron 450MHz CPU aboard the robots.

The 5dpo-2000 team for the RoboCup 2002 uses the same hardware at a higher resolution of 384x288, still at 15 bits of color.

The presented algorithm allowed that there was only a 30% increase in the number of pixels sampled and the time for the whole image processing is still only 11 ms (roughly a 60% increase). This execution time is for the full algorithm with weighed centers of mass calculations. If this feature is removed something like 2 ms may be gained.

5 Conclusion

The method presented is adequate to recent technological issues. It is also suited to Real Time applications because it is possible to have a likely estimate for the execution time shown to be quite short.

The main advantage of the method is that in the far region, the effective resolution is doubled and thus objects far away are more clearly seen.

It should be nonetheless clearly stated that the method involves re-sampling the image. This is interesting if computing power is not enough to process the whole image at maximum resolution.

One of the core issues of the method is generating offline the Dither Bitmap. To achieve this, camera height and inclination must be known and constant. If camera parameters change the method is valid only if a different Dither Bitmap is created.

The Dither Bitmap greatly affects execution time. The densities used should be validated having in mind the minimum object size and required precision in the measurements.

6 Future Work

More work still has to be done to evaluate correctly the importance of some additional issues like:

- Adapt the algorithm for moving cameras
- Measurement's accuracy can be improved by rescanning the boundaries of the objects at full resolution
- Other considerations for distance measurement may be taken into consideration

7 References

Eugene Hecht (1987) - *Optics*, 2nd Ed, Addison Wesley

Floyd *et al* (1976) - R.W. Floyd, L. Steinberg: *An Adaptive Algorithm for Spatial Grey Scale*, Proc. SID, 1976, Vol 17(2), 75-77

Zeggel *et al* (1993) - Thomas Zeggel, Olof Bryngdahl: *Error diffusion on an adaptive raster*. Electronic publishing, vol. 6(3), 213–218 (september 1993)

Moreira *et al* (2001) - *Vision Based Real-Time Localization Of Multiple Mobile Robots* António Moreira, Armando Sousa, Paulo Costa - 3rd Int. Conf. on Field and Service Robotics - June 11-13, 2001 Helsinki, Finland pp 103-106