# Speeding up probabilistic inference of camera orientation by function approximation and grid masking

Nicolau Leal Werneck
Universidade de São Paulo
Av. Prof. Luciano Gualberto tv. 3, 158
05508-900 São Paulo, SP, Brazil
nwerneck@usp.br

Anna Helena Reali Costa
Universidade de São Paulo
Av. Prof. Luciano Gualberto tv. 3, 158
05508-900 São Paulo, SP, Brazil
anna.reali@poli.usp.br

## ABSTRACT

This article presents modifications to an existing technique for camera orientation estimation intending to make it faster for use in real time applications and also for analysis of large image sets. The technique is based on likelihood maximization of a probability function that has the image gradient as the observed data and the camera orientation as parameter values. The camera orientation is inferred from the vanishing points of the image, and the directions of the edges in the environment are assumed to be in three mutually orthogonal directions. The first proposed modification is to substitute the expression that is calculated at each pixel by a computationally lighter approximation. The second proposal is to take in consideration only a few of the pixel lines and columns of the image during the calculations, performing a grid windowing of the image. This article presents the derivation and reinterpretation of the likelihood function approximation and also a performance evaluation.

## Keywords

Vanishing point, grid masking, camera orientation, camera localization, Bayesian inference, ML estimation.

## 1. INTRODUCTION

Camera localization is the Computer Vision problem of inferring the position and orientation of a camera in an environment from one or more pictures captured by it. Camera localization problems are defined by their different restrictions, specially the available data and what parameters are to be estimated. As usual in Computer Vision, it is an ill-posed problem of parameter estimation, and solutions are often based on procedures such as non-linear regression [SW89] and robust estimation [CKY09, HZ03]. One specific case of the localization problem is to estimate just the camera orientation from a single image under the restriction known as "Manhattan World", or also "LEGO Land", that the edges in the environment are in the directions of the coordinate axes. This article presents modifications to existing techniques [CY03, DIM02, SD04, DEE08] that solve this problem using the Likelihood Maximization principle, with a probabilistic observation model where the observed data is the image gradient, and the parameters to be estimated define the camera orientation in the world reference frame. Two modifications are proposed: the substitution of the expression calculated at each pixel by a simpler one, and the use of a grid mask to select pixels. The alternative expression caused great speed gains (60 fold in one test) while exhibiting good convergence. The subsampling technique also caused a 10 fold speed increase with just a 10% reduction of convergence probability in another experiment.



The proposed simplified expression can be seen as the result of a windowing operation by a mask that is calculated from the image gradient norm using a sigmoid function. While the original expression is strictly probabilistic, the proposal is similar to techniques such as Fuzzy Logic and Neural Networks.

In the remainder of this section the problem is further described and previous techniques are briefly reviewed. In Section 2 the existing techniques on which this proposal is based are better explained, and so is the developed technique. This section also brings results of experiments conducted with a database of images with solved orientation parameters to evaluate the proposal. Section 3 brings a few conclusions.

### 1.1 Problem geometry

The aim of the proposed technique is to obtain an estimate of the spatial orientation of a camera from a single image captured by it. The camera follows the simple *pinhole model* [TV98, chap. 2][HZ03, chap.

6]. In this model there is a camera reference frame whose origin is the focal point of the camera, and the image plane is located at $z = f$, where z is the direction that points outwards from the camera and into the scene. Image coordinates can be converted to this camera reference frame by a linear transformation involving the pixel size and the location of the image origin. The color of pixels are determined by the color of objects that are projected onto the image plane according to the classic perspective transformation [TV98]. The environment is assumed to be composed of rectangular parallelepipeds of faces with different colors and with edges aligned to the coordinate axes of the world reference frame.

The projections of the edges of these objects create edges on the captured images. These image edges typically produce gradient vectors with high magnitude that point to the direction orthogonal to the edge. Image gradients are approximately calculated by linear filters such as the one used in the well known Sobel detector [TV98, chap. 4]. In the present research the Scharr filter was employed [WS02]. The perspective projection causes the well known effect of producing *vanishing points* in the image. Lines that point to the same direction in the environment create either parallel lines in the image, or lines that converge to a vanishing point. The spatial orientation of the camera determines the position of the vanishing points, and the orientation can be therefore estimated from the directions of the image edges [HZ03, sec. 8.6]. This principle is the basis of many different techniques to estimate camera orientation.

## 1.2 Existing techniques

Many of the existing techniques for vanishing point or camera orientation estimation are either based on the Hough Transform [Shu99, CJRZ10] or on robust estimators [Tar09, Fö10] for matching edges extracted from the image and perform the desired estimation. Extracting edges and defining parameter space accumulators can be a nuisance for some applications, and this is one of the main reasons to look for alternative techniques. It is usually hard to extract edges with good precision, and also to match the edges that refer to the same direction. The technique presented in this article is part of a family of techniques that avoid these problems by using probabilistic models to infer camera orientation directly from pixel values, exploiting the vanishing point restriction.

The probabilistic model is used to perform *maximum likelihood* (ML) estimation to determine the camera orientation $\vec{\Psi}$ from a given input image. The differences between these similar techniques lie in the expression used for the calculation of the image likelihood given $\vec{\Psi}$ and the pixel values, more specifically the image gradient, and in the optimization procedure employed to find the optimal $\vec{\Psi}^*$ that maximizes the likelihood expression.

The directions of the environment edges must be known in order to infer camera orientation from vanishing points. In the present research the orientations are assumed to be in the directions of the coordinate axes, so the edge directions in camera coordinates are easily calculated from the rotation matrix that gives the camera orientation in relation to the world reference frame. Other than camera orientation, most of these techniques can be modified for other tasks such as discovering vanishing points in unknown directions and also estimating intrinsic camera parameters such as the focal distance $f$.

# 2. METHODOLOGY

This section brings more detailed explanations about how the existing and the proposed techniques work. They are all procedures that create an estimate of a camera orientation $\vec{\Psi}$ from a given input image. This orientation is a rotation matrix in three dimensions, and as such can be parameterized in different ways. The most popular alternatives are Euler angles, the Rodrigues formula, and quaternions, which are used in this work. But this representation is not relevant for the following subsections, where the reader can just assume $\vec{\Psi}$ is given as a 3D rotation matrix.

The next subsection describes the original probabilistic technique for estimating camera orientation from image gradient [CY03] and some modifications. The following subsection brings the new proposals. These techniques are all based on the Maximum Likelihood principle. They are more specifically *maximum a posteriori* (MAP) estimators, that can be seen as regularized ML estimators. These methods need a function called observation model, which is a conditional *probability density function* (PDF) of observing a measured data set given certain condition parameters. This function is used as a likelihood function, where the observed data is taken from the image gradient, and the conditional parameters are the camera orientation (related to the vanishing points locations), image coordinates of each pixel, and a pixel class that will be explained below. Once the expression is defined and the data collected, an optimization technique is used to find the parameters that maximize this MAP estimator. The $\vec{\Psi}^*$ found by this optimization is the desired camera orientation estimate.

## 2.1 Original observation model

In the first observation model proposed related to our technique [CY03] the likelihood of the whole image is factored as the product of the likelihoods of the

gradients $\vec{E}_{\vec{u}}$ at each pixel $\vec{u}$. These individual PDF are also further factored as products of the likelihoods of the gradient norms $E_{\vec{u}}$ and edge angles $\phi_{\vec{u}}$ yielding

$$P(\vec{E}_{\vec{u}}|m_{\vec{u}}, \vec{\Psi}, \vec{u}) = P(E_{\vec{u}}|m_{\vec{u}})P(\phi_{\vec{u}}|m_{\vec{u}}, \vec{\Psi}, \vec{u}). \quad (1)$$

The edge angle $\phi_{\vec{u}}$ is orthogonal to the gradient direction $\angle\vec{E}_{\vec{u}}$. The formula has two important characteristics. The first is that the PDF of the gradient norm $E_{\vec{u}}$ depends only on the pixel class $m_{\vec{u}}$. This class can be one of five possibilities: class 1 means the pixel is not an edge, classes 2–4 are edges on each of the three coordinate axes of the world reference frame and class 5 is a non-aligned edge.

The second characteristic is that the PDF of $\phi_{\vec{u}}$ also depends on $m_{\vec{u}}$, but also on the camera orientation $\vec{\Psi}$ and the coordinates of pixel $\vec{u}$. When $m_{\vec{u}} = 1$ or 5 we assume all gradient directions are equally probable, so $P(\phi_{\vec{u}}|m_{\vec{u}}, \vec{\Psi}, \vec{u})$ becomes a uniform distribution in these cases. For $m_{\vec{u}} = 2$, 3 or 4 we calculate the probability of the measured direction. For that we first use $\vec{\Psi}$ to calculate $\vec{r}^m = (r_x^m, r_y^m, r_z^m)$, a vector in the direction of the edges of the class $m_{\vec{u}}$ in the camera reference frame. The location where a line extended from $\vec{r}^m$ crosses the image plane is the vanishing point. The vector can also be parallel to the plane, in which case there is no actual vanishing point but it is still possible to calculate the directions of the edges. The direction on each pixel is:

$$\vec{\theta}_{\vec{u}}^m = \left(\frac{r_x^m}{r_z^m}f + u_x, \frac{r_y^m}{r_z^m}f + u_y\right). \quad (2)$$

One way to calculate $P(\phi_{\vec{u}}|m_{\vec{u}}, \vec{\Psi}, \vec{u})$ used in previous techniques is to determine the vanishing point direction angle $\angle\vec{\theta}_{\vec{u}}^m$, then subtract it from the edge direction $\phi_{\vec{u}}$, and this difference is then used as parameter to the PDF of the observation error in the measured edge directions. This PDF has been assumed in previous works to be uniform [CY03], triangular [DIM02], Gaussian [SD04] and a Generalized Laplace distribution [DEE08].

Two different PDF are used to implement $P(E_{\vec{u}}|m_{\vec{u}})$. For $m_{\vec{u}} = 1$, $P_{off}(E_{\vec{u}})$ is used, and $P_{on}(E_{\vec{u}})$ is used otherwise. Different assumptions have been made about these functions too. Both measured values [CY03, DIM02] and Gaussian models [SD04] have already been used.

As previously mentioned, the likelihood of the complete image is a product of terms given by Equation 1. This product can be used to define a ML estimator, but what is usually done is to improve it by using the information of *a priori* probabilities of $P(m_{\vec{u}})$, to define a MAP estimator. The logarithm of the resulting expression is also taken to replace the product by a summation, what does not change the location of the maximal points. Considering all this, and using $M^k$ for $P(m_{\vec{u}} = k)$, $\Phi^k$ for $P(\phi_{\vec{u}}|m_{\vec{u}} = k, \vec{\Psi}, \vec{u})$ we arrive at the expression:

$$L\left(\vec{\Psi}\right) = \sum_{\vec{u}} \log\Big(P_{off}(E_{\vec{u}})\Phi^1 M^1 + P_{on}(E_{\vec{u}})\Phi^5 M^5 + P_{on}(E_{\vec{u}})\textstyle\sum_{k=2}^{4}\Phi^k M^k\Big) \quad (3)$$

The camera orientation estimate is therefore the rotation $\vec{\Psi}^*$ that maximizes the function $L$. In the original proposal the summation is performed over all the image pixels [CY03], but just like with the PDF definitions, other researchers have proposed different ways to select subsets of the image pixels over which the summation should be performed, hoping to make the calculation faster and also smooth the estimator function. One proposal is to divide the image in square tiles, and sample a single pixel randomly from each one, a different pixel at each calculation [DIM02]. Another possibility is to select only a few of the pixels with the largest values of $E_{\vec{u}}$ [SD04]. The probabilistic modeling of the pixel being or not on an edge can be even dropped and substituted by the use of an edge-finding algorithm [DEE08]. In this case the argument of the log in Equation 3 becomes simply $\sum_{k=2}^{5}\Phi^k M^k$, but this technique depends on an initial edge extraction, that did not exist in the original proposal.

Other aspects where the techniques differ is the application of the Expectation-Maximization algorithm, where values for $M^k$ are also iteratively estimated [SD04, DEE08], and the optimization algorithms used. Alternatives range from coarse-to-fine search at regularly sampled points [CY03], stochastic importance sampling [DIM02], and continuous non-linear optimization methods [SW89] such as Levenberg-Marquardt [SD04] and BFGS [DEE08].

## 2.2 Function approximation

This subsection describes the first major modification investigated in this research, which is substituting the original arithmetical expression for the likelihood function by a computationally simpler approximation. The following subsection covers the use of a grid mask to select the pixels to be considered in the calculations.

Tests performed with an implementation of the original likelihood expression (Equation 3) revealed that much of the computation time was spent on functions to compute the logarithm and the arc-tangent used to calculate $\angle\vec{\theta}_{\vec{u}}^m$. A removal from the program of the procedure calls related to these operations, while keeping all the rest of the

calculations, resulted in an approximately ten fold speed gain, showing experimentally that avoiding these operations can be a good strategy to reduce the calculation time. Arc-tangent was the most costly operation of the three, considering both the time of a single calculation and the number of calls at each calculation iteration in the summation loop.

The modifications begin by replacing the logarithm with the first-order approximation

$$\log(b+a) \approx \frac{a}{b} + \log(b), \tag{4}$$

where $a$ represents the terms that depend on $\vec{\Psi}$, and $b$ the terms that remain constant during the optimization. The $\log(b)$ term can therefore be ignored as it does not influence the solution, and the resulting approximation becomes

$$\sum_{\vec{u}} W'(E_{\vec{u}}) \sum_{k=2}^{4} \Phi^k \frac{M^k}{\Phi^1}, \tag{5}$$

where the *mask generating function*

$$W'(E_{\vec{u}}) = \left( \frac{P_{off}(E_{\vec{u}})}{P_{on}(E_{\vec{u}})} M^1 + M^5 \right)^{-1}. \tag{6}$$

The function $W'$ produces, at least with the appropriate parameters, a sigmoid curve, similar to the logistic or to the hyperbolic tangent functions. The second approximation used was to replace this function by $W$, the logistic function applied to $E_{\vec{u}}$ translated by $p_1$ and scaled by $p_2$

$$W(E_{\vec{u}}) = \left( 1 + e^{-p_2(E_{\vec{u}} - p_1)} \right)^{-1}. \tag{7}$$

Replacing $W'$ for $W$ at Equation 5 and ignoring the constant $M^k/\Phi^1$, that only scales the function, finally produces the proposed estimator:

$$\tilde{L}\left(\vec{\Psi}\right) = \sum_{\vec{u}} W(E_{\vec{u}}) \sum_{k=2}^{4} \Phi^k, \tag{8}$$

Figure 1 displays, at the top, the probability models of the gradient magnitudes with measured values, provided by the authors of [CY03], and also the Gaussian models from [SD04] (mean 8.28 and standard deviation 6.21 for $P_{on}$, and respectively 1.13 and 0.77 for $P_{off}$). On the bottom of the figure, the continuous and dashed curves are $W'$ obtained from the two PDF models mentioned, and the red dotted curves are $W$ with two different sets of parameters ($p_1 = 10$ $p_2 = 0.4$ and $p_1 = 3.1$ $p_2 = 3.0$).

Figure 2 shows an image from the YorkUrbanDB image database [DEE08]. This image set has 102 indoor and outdoor images of man-made environments, and the orientation of each image was

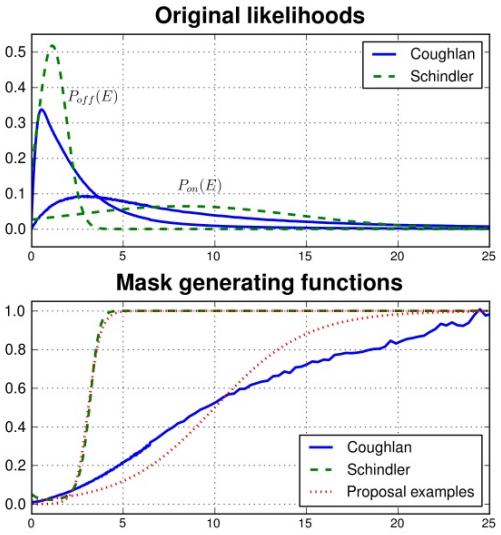


Figure 1: Original gradient magnitude likelihood functions, resulting mask generating functions and examples of the proposed function.

obtained from edges and with a manual labeling process. Intrinsic parameters of the camera are also provided, enabling interested researchers to test their techniques and compare to others. Possible radial distortions of the images were not taken in account in this work, but the projection center coordinates and focal distance that are provided were used.

The leftmost graphic of the figure displays the input image. The next one displays the values of $W$ calculated over each pixel, with white representing the zero level, ($p_1 = 20$ and $p_2 = 0.2$ were used). The two graphics to the right display the horizontal and vertical components of the normalized direction vector. The red color denotes negative values, but even in a monochromatic mode it is possible to see how edges in the direction of the derivative vanish on each graphic. The edge mask obtained with $W$ has been applied to these gradient images, clearing out the noise that would be otherwise noticeable in the large white areas of these images.

In the program created to implement this expression the edge mask is calculated and stored in memory before the optimization procedure starts, so only memory accesses are needed to obtain the values during the calculations. Something similar can be done with other techniques, because $P(E_{\vec{u}}|m)$ does not depend on $\Psi$, only $\Phi^m$ does.

The last modification done to the likelihood expression was to substitute the calculations of arc-tangents by dot products. Instead of calculating the angles of the gradient and vanishing point directions, these vectors are simply normalized and multiplied by each other. Because the gradient is orthogonal to the edge direction, this multiplication

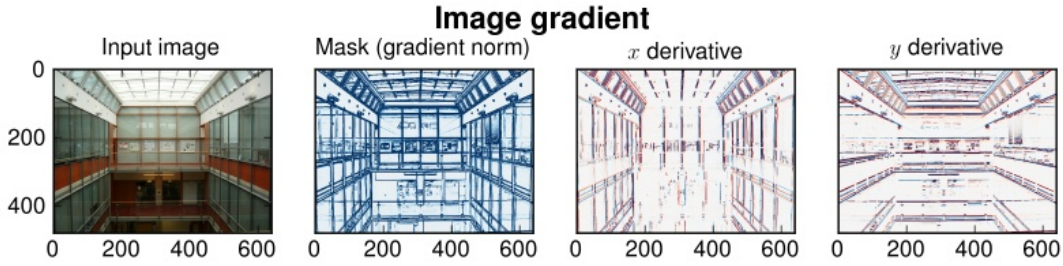


Figure 2: Gradient of an YorkUrbanDB image. The second image is the edge mask calculated from the gradient vectors absolute values. The two rightmost graphics are the masked gradient $x$ and $y$ components.

yields $\gamma^m = \sin(\phi_{\vec{u}} - \angle\vec{\theta}^m_{\vec{u}})$. This is a good approximation of the identity function for small values, so the product result can be directly used in the angle error PDF. The function used was therefore:

$$\Phi^m = \begin{cases} \frac{2}{p_3}\left(1 - \frac{|\gamma^m|}{p_3}\right) & \text{if } |\gamma^m| < p_3 \\ 0 & \text{if } |\gamma^m| \geq p_3 \end{cases}, \quad (9)$$

where we note that the $2/p_3$ multiplication can be dismissed without affecting the optimization results.

The normalization of $\vec{E}_{\vec{u}}$ and $\angle\vec{\theta}^m_{\vec{u}}$ can be performed quickly using a special `rsqrt` instruction available in many modern processors that calculates an approximation of the reciprocal of the square root of numbers. This instruction was used in the implementation tested, and so were SIMD (*single instruction, multiple data*) instructions that allow calculations to be performed simultaneously both for the three vanishing points, and also for the three image channels when possible. The three image channels were independently considered in the calculations, with just the pixel coordinates and $\vec{\theta}_{\vec{u}}$ in common. The final likelihood value is therefore the summation of likelihoods for each channel.

The program was implemented using `Cython` [Sel09], with a few routines implemented in C in order to make use of the special processor instructions mentioned. Another implementation was made based on [CY03], using arc-tangent and logarithm calls inside the loop, but with some similarities to the implementation of the proposal, such as using SIMD instructions for some operations, and caching constant values.

Tests were performed with the YorkUrbanDB images at different values of $\Psi$ to measure the speed of the proposed function relative to this implementation of the original. Speed gains from $50$ up to $64$ times were found in one computer (`c1.xlarge` instance from Amazon Web Services [Ser]), where the mean time to calculate the likelihood of one image using the classic function was $1.10 \pm 0.06$s versus $18.9 \pm 2.4$ms for the proposed algorithm. Although these numbers naturally varied according to the processor employed, accelerations of more than 10 times were often detected in other tests.

The positive impact of these function modifications on the calculation speed is not surprising. But the impact of these modifications on the performance of the optimization procedure must be now studied to validate the proposed technique. This analysis will be presented in Subsection 2.4. But it should be noted that this proposed modification did not intend to numerically approximate the original likelihood function values. The original function serves more as a theoretical foundation, and the modifications do not seek to approximate it exactly, but only retain characteristics such as the positions of the extremal points and gradient directions.

When the logarithm of the likelihood is used instead of the original function in an optimization, the produced function does not approximate the original numerically, but is still useful for the optimization. So the performance of such modifications should not be measured by looking at approximations errors, but at the optimization results instead. In the same way, because the modifications proposed here include dropping some constant terms, the resulting function cannot be compared to the original function, so no error analysis was performed, only performance analysis of the optimization procedure. Despite of that, the modifications are in fact initially based on first-order approximations of the original function, justifying the use of the term *approximation*, even though the final proposed function does not approximate the original one numerically.

The proposed function also differs from the original in that the parameters of the mask generating function are only indirectly related to the gradient norm probabilities. While it is possible to fit the parameters to a mask function taken from histograms, it is better to look for parameters that maximize the performance of the final optimization procedure. The sensitivity of the performance to these parameters, and also to the gradient norm probabilities is a topic that the proposed modifications bring up, but was not studied here.

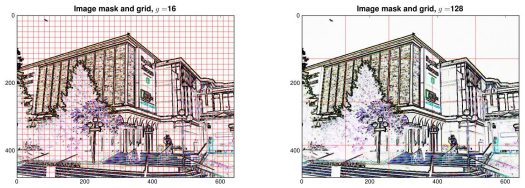


Figure 3: Two example grids with spacing equal to 16 and 128 over the norm of the gradient of an example image.

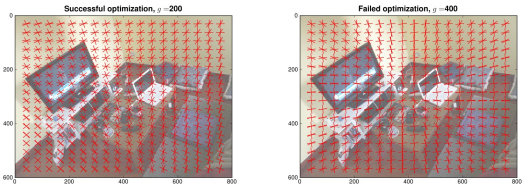


Figure 4: Successful and failed optimizations.

## 2.3 Grid mask

A sampling strategy was introduced to reduce the number of pixels used in the calculations, and also try to make the estimator smoother. The strategy is to select only a few of the pixel lines and columns of the image. These were selected at regular intervals, with the spacing controlled by the parameter $g$. The result is the same of applying a mask to the image with the shape of a square grid, with a continuous line or column at every $g$ pixels, starting from the image origin. When $g = 1$ all pixels are used. When $g = 2$, $3/4$ of the pixels are used, and $7/16$ when $g = 4$.

The grid masking is proposed here as a theoretically more suited way to subsample images when edges are the target features. Other subsampling techniques simply regard edges as a kind of pixel, and not as a geometric entity without area. Some authors quote statistics such as "10% of the image pixels are edges", but such statements miss some important points about image edges. The number of pixels of an image increases with the square of resolution, but the number of pixels that lie over and edge should increase linearly. New edges may be introduced as resolution increases, affecting positively this proportion of edge pixels to image size, but the relative number of pixels of an existing edge still decreases linearly with resolution, and subsampling strategies should take this effect in consideration.

As image resolution increases the number of edge pixels found over a grid line or column should remain constant, while non-edge pixels increase linearly with resolution — assuming that no new edges are introduced, and that edge directions are not exactly aligned to the grid. One interesting characteristic of grid masking is that if the edges have a minimum length, the grid spacing can be made small enough as to guarantee that a minimum number of points over any edge in the images is sampled. Grid masking also avoids sampling groups of neighboring pixels, what is generally thought to be good because pixels are assumed to be independent in the probabilistic model. Figure 3 shows the grid lines and columns overlaid to the gradient of the three channels of an image from the YorkUrbanDB database. The top graphic has the grid spacing parameter $g = 16$, and the lower $g = 128$.

## 2.4 Optimization

With the likelihood function and sub-sampling technique defined, an optimization technique can now be used to produce orientation estimates from input images. The algorithm used was the modified Powell's method from `SciPy` [JOP$^+$ ]. Figure 4 displays a successful optimization, obtained with $g = 200$, and a failed one with $g = 400$. This is an 800x600 pixels image captured with a consumer digital camera. Line segments in the directions of the three vanishing points obtained form the solution were plotted in regularly spaced points over the images, and it is possible to see how the edges are aligned to the objects in the environment. In the failed optimization the solution found was not much far from the initial condition, which was no rotation.

The parameterization used for the rotations was quaternions. The vector $\vec{\Psi}$ has three dimensions, and are the three quaternion parameters that are directly related to the direction of the rotation axis. The fourth parameter, related to the rotation angle, is calculated as $\sqrt{1 - ||\vec{\Psi}||^2}$. If $||\vec{\Psi}|| > 1$ no quaternion can be directly produced. In this case the quaternion is obtained from $-\vec{\Psi}/||\vec{\Psi}||$. It should be noted that no symmetries were taken in account in this parameterization, so multiple $\vec{\Psi}$ values are equally acceptable solutions, and can be obtained from each other by 90 degrees rotations around the axes. The problem of associating the axes properly, when possible, was not considered in this research.

As in previous works, optimization is initiated from different starting points [DEE08], although only two were used in the present experiments. One point considers no rotation, and the other a 45 degrees rotation around the vertical axis. This explores the tendency of the camera to be upright, and the ambiguity resulting from 90 degree turns. After the two optimizations are performed, the solution with the highest likelihood is picked as the best estimate.

Figure 5 shows an evaluation of this optimization for different grid spacings $g$. The parameters used for $\tilde{L}$ in this experiment were $p_1 = 20$, $p_2 = 0.2$ and $p_3 = 0.1$. There is a compromise between calculation speed and the quality of the solutions obtained. The decreasing green curves show the probability $p$, estimated from the N=102 images, of the obtained

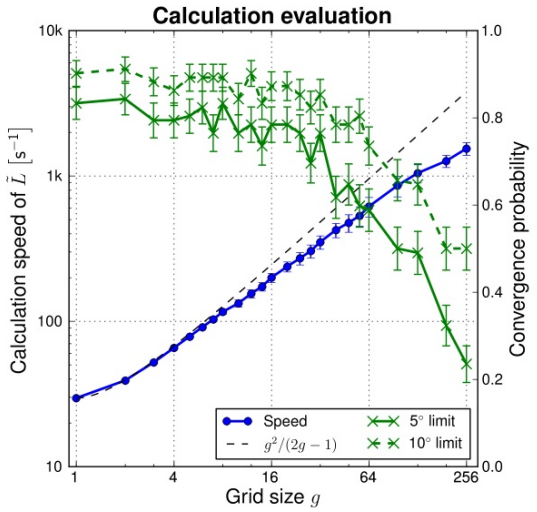


Figure 5: Speed of the calculations as a function of the grid spacing, and two quality evaluations.

solution lying at $10^\circ$ or $5^\circ$ of the orientation provided in the database [DEE08]. The $10^\circ$ curve is naturally above the $5^\circ$ one. The uncertainty interval plotted is a single standard deviation above and below the points, calculated as $\sqrt{p(1-p)/N}$.

The speed estimate is the number of times $\tilde{L}$ can be calculated per second as a function of $g$. To calculate this, the elapsed time and number of function calculations performed were first stored for both optimization runs of all test images. The speed was then estimated for each optimization run by dividing the number of calculations by the elapsed time. The mean number of iterations was 276 for all optimization trials with all $g$ values, with a 89.3 standard deviation. The increasing blue continuous line in the graphic is the mean and $6\sigma$ interval of this speed for all optimizations performed for each $g$. It should be noted this experiment was performed in a slower machine compared to the one used for the measurement reported in Subsection 2.2.

The increasing black dashed line in Figure 5 shows how speed should increase if it depended only on the reduction of the number of pixels, where speed gain should be $g^2/(2g-1)$ . The smaller speed values that were measured are coherent with the addition of a constant time to the calculation time, the reciprocal of the speed value.

This analysis only considers the individual performance of the proposed target function and the effects of the grid sampling. Another test was performed in order to validate the proposed function. The objective was to find out if the modifications were causing the extremal points to be located in positions further from the true solutions than the points produced by the original function.

To perform this test the solution found with the proposed method was used as initial estimate for a second optimization on the original function. The error of the first and the second optimizations compared to the estimate in the database were then analyzed. The modifications would be considered destructive if the errors in the first optimization were higher than the errors from the second, i.e. the second optimization would "fix" the first one. On the other hand, if the modifications are acceptable the second optimization should not improve the solutions much.

The result was that from the 102 YorkUrbanDB images 53 had their errors reduced after the second optimization. From these, 5 were improvements from more to less than $10^\circ$ away from the correct solution. On the other hand, from the 49 cases where the second optimization ended with a larger error, there were 6 cases where the initial solution was below $10^\circ$ but the second was beyond. So there is no indication that using the original expression can be critical to improve the performance obtained with the proposed function, at least with the optimization algorithm that was used and with no subsampling performed.

## 3. CONCLUSION

This article demonstrated modifications made to existing techniques for camera orientation estimation to attain higher calculation speed. The techniques are based on the optimization of a MAP estimator that has the image gradient values as observed data, and the camera orientation as estimated parameter. It works by finding the orientation that causes the best alignment of the image gradient to the vanishing points created by the directions of the three mutually orthogonal axes of the world reference frame.

The original expression to calculate the likelihood was modified by an approximation that avoids the calculation of arc-tangents by using dot products, and also replaces the logarithm of a summation at the expression for each pixel by a summation where all the terms are strictly dependent on the gradient directions and camera orientation. These pixel summations are weighed in the total image summation by a coefficient calculated by applying a sigmoid function to the gradient norms.

This coefficient takes the role performed originally by the likelihoods $P_{on}$ and $P_{off}$, and also the *a priori* probabilities $M^1$ and $M^5$. The need to measure these parameters is replaced by having to choose just $p_1$ and $p_2$. The third parameter $p_3$ shapes the likelihood of gradient directions. More tests still have to be conducted to determine the best parameters, but the technique seems to be robust to variations on them. Outside of these parameters, the other parameters that

must be set in order to use the technique are the ones related to the optimization.

A grid masking technique was also proposed to select a subset of the image pixels to take in consideration in the calculations. It was inspired in the usual curve tracking technique of searching for edges over spaced lines normal to an initial estimate of the curve location[BI98, chap.5 ], and also on the Canny edge extractor [TV98]. It subsamples the image in a deterministic and more reliable way, and has been proven effective.

Some planned extensions to this research are to better choose the function parameter values and turn the grid masking into a search of maximal points of the derivative in the direction of the line or column. The gradient calculations can also be restricted to the grid vicinity to speed up calculations. Other subsampling techniques can also be applied together with a grid mask. For example, random sampling could be performed only within the mask pixels, or a random sampling could be performed in the whole image initially, but instead of picking just a single pixel from each trial, picking a whole group of pixels inside a cross or square mask centered at each generated pixel.

This fast orientation estimation algorithm is planned to be used in real time to track the orientation of a camera with a Kalman filter or a similar technique. An attempt will be made to reuse the data remaining from the grid masking to also extract edges. The resulting edge observations will be fed to a monocular simultaneous localization and mapping (SLAM) system [NDL08] that exploits the restrictions on the edge directions.

## ACKNOWLEDGEMENTS

The authors thank the support from Capes, CNPq (Proc. N. 475690/2008-7 and N. 305512/2008-0) and FAPESP (Proc. N. 2008/03995-5).

## REFERENCES

[BI98] Andrew Blake and Michael Isard. *Active Contours*. Springer, 1998.

[CJRZ10] Xuehui Chen, Ruiqing Jia, Hui Ren, and Yinbin Zhang. A new vanishing point detection algorithm based on hough transform. *Computational Sciences and Optimization, International Joint Conference on*, 2:440–443, 2010. doi:http://doi.ieeecomputersociety.org/10.1109/CSO.2010.163.

[CKY09] Sunglok Choi, Taemin Kim, and Wonpil Yu. Performance evaluation of RANSAC family. In *20th British Machine Vision Conference*, 2009. Available from: http://www.bmva.org/bmvc/2009/Papers/Paper355/Paper355.pdf.

[CY03] James Coughlan and Alan Yuille. Manhattan world: orientation and outlier detection by bayesian inference. *Neural Comput.*, 15(5):1063–1088, 2003. Available from: doi:10.1162/089976603765202668.

[DEE08] Patrick Denis, James H. Elder, and Francisco J. Estrada. Efficient edge-based methods for estimating manhattan frames in urban imagery. In David A. Forsyth, Philip H. S. Torr, and Andrew Zisserman, editors, *ECCV (2)*, volume 5303 of *Lecture Notes in Computer Science*, pages 197–210. Springer, 2008.

[DIM02] Jonathan Deutscher, Michael Isard, and John Maccormick. Automatic camera calibration from a single manhattan image. In *Eur. Conf. on Computer Vision (ECCV)*, pages 175–205, 2002.

[Fö10] Wolfgang Förstner. Optimal Vanishing Point Detection and Rotation Estimation of Single Images of a Legolandscene. In *Int. Archives of Photogrammetry and Remote Sensing*, pages 157–162. ISPRS Symposium Comm. III, Paris, 2010.

[HZ03] Richard Hartley and Andrew Zisserman. *Multiple View Geometry*. Cambridge Univ. Press, 2nd edition, 2003.

[JOP+ ] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. Available from: http://www.scipy.org/.

[NDL08] José Neira, Andrew J. Davison, and John J. Leonard. Guest editorial special issue on visual slam. *Robotics, IEEE Transactions on*, 24(5):929 –931, oct. 2008. doi:10.1109/TRO.2008.2004620.

[SD04] Grant Schindler and Frank Dellaert. Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In *CVPR (1)*, pages 203–209, 2004.

[Sel09] Dag Sverre Seljebotn. Fast numerical computations with cython. In Gaël Varoquaux, Stéfan van der Walt, and Jarrod Millman, editors, *Proceedings of the 8th Python in Science Conference*, pages 15–22, Pasadena, CA USA, 2009. Available from: http://conference.scipy.org/proceedings/SciPy2009/paper_2.

[Ser] Amazon Web Services. Available from: http://aws.amazon.com.

[Shu99] Jefferey A. Shufelt. Performance evaluation and analysis of vanishing point detection techniques. *PAMI, IEEE Transactions on*, 21(3):282–288, 1999. doi:10.1109/34.754631.

[SW89] George A. F. Seber and Christopher J. Wild. *Nonlinear Regression*. John Wiley & Sons, Inc., 1989.

[Tar09] Jean-Philippe Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1250 –1257, sep. 2009.

[TV98] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.

[WS02] Joachim Weickert and Hanno Scharr. A scheme for coherence-enhancing diffusion filtering with optimized rotation invariance. *Journal of Visual Communication and Image Representation*, 13(1-2):103 – 118, 2002. Available from: doi:10.1006/jvci.2001.0495.