CEIG - Spanish Computer Graphics Conference (2014)
Pere-Pau Vázquez and Adolfo Muñoz (Editors)

Depth from a single image through user interaction

A. Lopez[1], E. Garces[2], D. Gutierrez[2]

[1]Institute of New Imaging Technologies, Universitat Jaume I, Spain
[2]Graphics and Imaging Lab, Universidad de Zaragoza, Spain

Abstract

In this paper we present a method to obtain a depth map from a single image of a scene by exploiting both image content and user interaction. Assuming that regions with low gradients will have similar depth values, we formulate the problem as an optimization process across a graph, where pixels are considered as nodes and edges between neighbouring pixels are assigned weights based on the image gradient. Starting from a number of user-defined constraints, depth values are propagated between highly connected nodes i.e. with small gradients. Such constraints include, for example, depth equalities and inequalities between pairs of pixels, and may include some information about perspective. This framework provides a depth map of the scene, which is useful for a number of applications.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms, I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Depth cues

1. Introduction

Stereoscopic cinema has received growing interest in the last decade. While its production for animated CG movies is straightforward because all the scene content is known in advance, it presents a real challenge for 2D films where 3D structure is coupled into the image pixels. Recent advances in 3D image capture with stereo cameras, multiple views [SCD*06] or light fields [KZP*13] aim to ease this task, although its use is limited to new cuts and increase costs and direction difficulty. Currently, most of the work to convert a 2D movie into 3D is done frame by frame by skilled artists.

Obtaining the depth map of the scene is the first step of this task and, due to the inherent ambiguity of single images, making this process fully automatic becomes an ill-posed problem. Learning-based methods [SSN09, HEH11, HB13] are a plausible solution for common scenarios such as landscapes or urban environments, although they are unable to generalize to more complex or unseen scenes i.e. scenarios which are not part of the training data. Other methods directly model the 3D geometry of the scenes [OCDD01, GIZ09] but they usually require too much interaction and skill from the user. A few recent methods [WLF*11, YSHSH13] try to exploit user knowledge through a less demanding interaction to obtain relative depth values instead of a 3D model. Our method belongs to this latter group where the goal is to obtain an approximated depth map relying on a sparse set of user strokes whose values are propagated through the scene. Contrarily to these works, our method can handle perspective constraints given by the user.

In particular, we aim to infer depth values from hints given by the user in two ways: via equality/inequality constraints or via the perspective tool. The first one consists on a set of points with similar/different depth, distributed throughout the scene. The latter consists on locating in the image the horizon line and the ground plane. We incorporate these cues into a flexible optimization framework, which propagates this information through the scene leveraging gradient information.

2. Related work

There are an increasing number of works that try to obtain the depth of a scene from a single image. We can classify them as automatic and semiautomatic methods.

Automatic methods. Methods that try to automatically obtain depth from a single image can be classified into two broad families [CC13]: learning-based and Gestalt-based approaches.

DOI: 10.2312/ceig.20141109

Learning-based approaches usually infer 3D values for a number of small regions obtained from image over-segmentation through supervised learning. Most of the approaches in this family rely on the quality of the detection stages (i.e. image segmentation). Saxena et al. [SSN09] use a Markov Random Field (MRF) to infer 3D position and orientation for a number of small homogeneous patches in the image. Hoiem et al. [HEH11] remove weak boundaries between regions, according to an occlusion confidence, whose strength is inferred from the ground truth data using a Conditional Random Field (CRF). Liu et al. [LGK10] first perform a semantic segmentation of the scene (through a learned multi-class image labeling MRF), and then they use the predicted semantic class labels to guide the 3D reconstruction, by enforcing class-related depth and geometry priors. Instead of semantic information, Haines and Calway [HC12] use texture and color information for detecting planes. They train a classifier to estimate plane orientation at each pixel, which is segmented into distinct regions in order to derive final planes. There are also example-based methods, like [HB13], which need a database of example 3D geometries. They search the database for those examples that look similar to a given patch, such that overlapping patches provide several depth estimates, which are combined to estimate each pixel's depth.

In general, learning-based approaches depend on the availability of ground truth data and their use is limited to the trained/collected image types. In addition, they rely on the results of a segmentation/labelling algorithms and most of these algorithms provide a partition of the image that ignores depth cues.

The second family, Gestalt-based approaches, pioneered by Nitzberg and Mumford [NM90], try to model depth perception mechanisms to estimate occlusion-based monocular depth. These methods are influenced by phycho-visual studies that describe T-junctions as fundamental atoms of monocular depth perception, and whose benefits at early stages were demonstrated by Caselles et al [CCM96]. A number of these approaches formulate depth estimation as a layer image representation problem. In general, T-junctions are extracted from an image partition, and analyzed to obtain a global depth ordering [ART10, GWZS07].

To overcome the hard decisions that may require to provide the partition that guides depth estimation, some works jointly perform image segmentation and depth estimation [Mai10, PS13]. Maire [Mai10] encodes affinity and ordering preferences in a common image representation, and proposes an algorithm that provides both image segmentation and a global figure/ground ordering on regions. Palou and Salembier [PS13] use a region-based hierarchical representation jointly with detection of special points, such as T-junctions and highly convex contours, to recover depth ordering.

In general, these methods provide a set of ordered layers, which are regions assumed to form planes parallel to the image plane. Combining segmentation and depth cues increments the computational burden, as segmentation is a difficult problem by itself. The work by Calderero and Caselles [CC13], however, avoids integrating segmentation into the framework: they first extract multi-scale low-level depth features to estimate depth order between neighbouring pixels, and then, integrate the local features to create a consistent depth order of the scene by applying a bilateral filter. Our method also avoids integrating segmentation into the framework, but we exploit user interaction, instead of low-level image features, which in turn, could be integrated in our framework in a future extension.

Semiautomatic methods. These methods exploit humans' ability to interpret 2D images, by requiring some input information from the user. Some methods face this problem by extending traditional photo editing tools to 3D [OCDD01]. Although this is the most straightforward solution, and allows to provide high accuracy and robustness to the 3D structure, it requires a high effort from the user –a lot of user interaction is needed– as well as certain level of skill, or at least some practice.

Other works exploit geometric information (lines, planes) in the scene. If the image contains enough perspective information, it is possible to obtain a 3D model of the objects from both user input and features like straight edges [LCZ99, CRZ00]. In these methods, the user must provide information about segments and points in different planes to compute homologies between planes, and consequently, the 3D structure. There are a few attempts to make this process fully automatic, for indoor [LHK09] and outdoor scenes [RB13], but they are limited to scenes that contain a great amount of straight lines to be able to infer 3D structure. There are also attempts to automatically reconstruct curved surfaces from a single view, like [PZF06], that can construct a 3D model of simple object shapes and views, using a-priori object class information.

In general, these methods are limited to simple objects or scenes with enough perspective information (like architectural scenes). We seek to obtain a simple 2.5D structure of the scene, that is, the depth corresponding to each pixel in the image, which is enough for a number of applications. The importance of applying human knowledge to computational problems that involve cognitive tasks is revealed in the work of Gingold et al [GSCO12], whose approach utilizes humans for finding depths, among other visual tasks. They apply human computation by decomposing the problem into micro-tasks, which are based on perception and solved via crowdsourcing, and whose solutions are combined afterwards to solve the proposed problem. Our approach is very different from this, and more similar to other methods that obtain a 2.5D structure of the scene guided by the user. The more relevant ones are described below.

Some of these methods have been developed to obtain

depth from drawings or cartoons [GIZ09, SSJ*10]. In this case, the problem becomes slightly simpler, as the input image is a line drawing with visible contours. Most approaches provide sketch-based 3D modelling tools to obtain free-form surfaces from 2D sketches like [GIZ09]. Sýkora et al [SSJ*10] formulate an optimization framework that tries to mimic the way a human reconstructs depth information from a single image. Our problem formulation is very similar to the initial formulation in [SSJ*10], which was previously used for image segmentation [Gra06]. Instead of propagating labels (or colors), Sýkora et al propagate depth values. However, they propose an approximation to lower computational overhead by decomposing the optimization problem into two separate steps: first, a multi-label segmentation tailored to cartoon images, and second, depth assignment via topological sorting, plus depth smoothing. Our method propagates depth values through image pixels without requiring a previous segmentation.

Another work that infers the 2.5D structure from user interaction on real images, uses image content to propagate the depths painted by the user [WLF*11]. They integrate a discontinuous warping technique into the framework to simultaneously fill disoccluded regions to obtain a valid stereo pair of images from a single view. The user marks absolute depth values, as sparse scribbles, and these are propagated to unknown parts using content-aware weights. The user can incrementally add scribbles to refine the results. Our approach is similar, but we expect the user to give hints instead of painting absolute depth values.

Transfusive image manipulation (TIM) [YJHS12] was used in this context by Yücer et al. [YSHSH13] to obtain a depth map from a single image. They require the user to specify pairs of scribbles that represent relative depth inequality constraints. Our algorithm is similar to theirs in that sense, however, we allow for additional constraints that leverage perspective cues.

3. Our method

We formulate the problem of estimating depth as a graph-based optimization problem with a given set of constraints. We require the user to assign at least one point in the furthest and one point in the nearest part of the scene. These points, together with any other constraint defined by the user, are integrated in the optimization process described below.

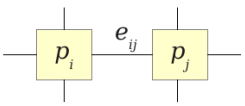

Figure 1: *Graph construction: Pixels as nodes and neighbourhood relationships as edges.*

3.1. Problem formulation

We represent the image $\mathcal{I}$ as an undirected graph $\mathcal{G} = (V, E)$, being $v_i \in V$ the set of vertices corresponding to pixels of the image, and $e_{ij} = (p_i, p_j) \in E$ the set of edges connecting pairs of neighbouring pixels in a 4-connected neighbourhood (see Figure 1). Each edge has a weight w_{ij}, which depends on the image gradient and defines the depth similarity between two connecting pixels as follows:

$$w_{ij} \propto e^{-\beta(\mathcal{I}_i - \mathcal{I}_j)^2} \tag{1}$$

where $\mathcal{I}_i$ represents the image intensity at pixel p_i, and β is a parameter of this method. This function maps intensity changes to edge weights, such that weight values are high in homogeneous areas; while they are low in presence of high intensity gradients, implying a possible depth discontinuity.

Our goal is to propagate depth values to all the pixels of the image given user constraints and the weights computed from Equation 1. We minimize the following function:

$$\begin{aligned} &\text{minimize:} && \sum_{\forall e_{ij} \in E} w_{ij}(d_i - d_j)^2 \\ &\text{subject to:} && d_k = \hat{d}_k, \quad \forall p_k \in S \end{aligned} \tag{2}$$

where d_i represents the depth value estimated at pixel p_i, S is the set of pixels p_k which are given a depth (just the maximum or minimum depth), which is denoted as $\hat{d}_k$. This formulation was developed by [Gra06] for image segmentation. Sýkora et al. [SSJ*10] used a simplified version of this formulation to assign depths to cartoons. In both cases, they assumed that a number of depth values (or seeds) are given or computed in a previous step. In our case, we require a minimal number of depth values of two: one point in the farthest and one point in the nearest region, as shown in Figure 2 (top-left).

This problem can also be formulated as a quadratic program:

$$\begin{aligned} &\text{minimize:} && d^T L d \\ &\text{subject to:} && d_k = \hat{d}_k, \quad \forall p_k \in S \end{aligned} \tag{3}$$

where d is the array of depths to be estimated, and L is a sparse large matrix that represents the Laplace-Beltrami operator [Gra06], which in turn $L = A^T W A$, where W is a $m \times m$ diagonal matrix ($m = |E|$) called the constitutive matrix, and A is called the incidence matrix. W contains the weights of each edge along the diagonal, and A stands for:

$$A_{e_{ij} p_k} = \begin{cases} +1 & \text{if } i = k, \\ -1 & \text{if } j = k, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

3.2. Weight values

Regarding the weight values, w_{ij}, a number of considerations must be taken into account. Our simpler solution

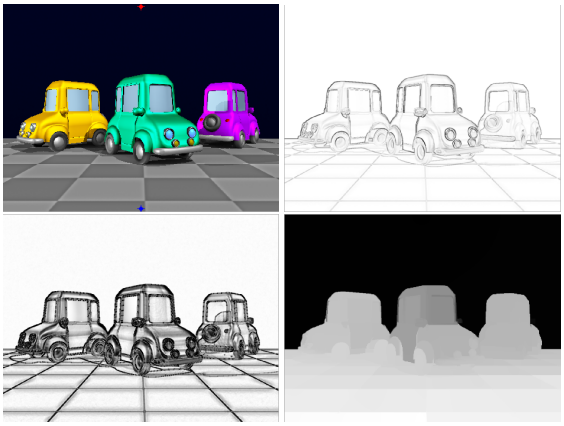

Figure 2: *Top-left, original image with two selected points: red in the far, blue in the near. Top-right, gradient on luminance channel of the CIELAB color space. Bottom-left, weights computed from gradient. Bottom-right, depths obtained from optimization with* $\beta = 5$ *and Sobel filters for gradient computation.*

uses the luminance channel of *CIELAB* color space to compute intensity values I_i (see Figure 2), although Equation 1 could be easily modified to handle the three color channels or any other vector value to compute the gradient. Notice that throughout this paper gradient magnitudes are shown negated: zero gradient is painted in white and the maximum gradient in black. Also, depths are considered inversely: maximum depth is coded as zero (black) and the minimum depth is coded white. In all the experiments, we use the Sobel operator to compute gradient values. We force w_{ij} to be in the range $[0,1]$, so that a gradient of zero would be represented with a weight $w_{ij} = 1$. In general, $w_{ij} = 0$ means non-existing neighbour relationship and should be avoided because it increases instability in the optimization. Therefore, we limit w_{ij} to the range $[w_{min}, 1]$, being $w_{min} = 0.001$ in all the experiments.

The optimization propagates the minimal user input from Figure 2, top-left, to the rest of the image to obtain depths, such that high intensity gradients prevent from propagation, while homogeneous areas obtain similar depths. This simple example illustrates how the approach works with minimal input. However, the obtained depth map can be improved considerably (the car in the middle should be closer than the other two) with additional input.

In Figure 3 we compare different depths obtained with different values of β. Intensity or color discontinuities (dark values in the center-left image) can be seen as barriers for propagation of depths (right column). Therefore, higher values of β will produce lower weights (darker values in the bottom-left image), and therefore taller barriers. The results in Figure 3 were obtained by taking into account the gradient

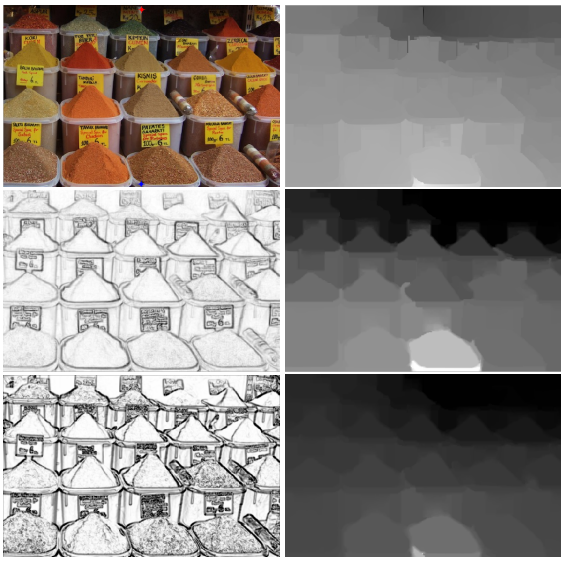

Figure 3: *Left column, from top to bottom: original image with two selected points (red in the far, blue in the near), image gradient (dark values represent intensity/color discontinuities) and weights for* $\beta = 5$*. Right column, depths obtained with different values of* β*: from top to bottom, 5, 10 and 20. Original image courtesy of Yotam Gingold.*

in both luminance and color, while Figure 2 was obtained with only the luminance channel. Also, in this example, a larger number of input seeds improves the resulting depths, as shown in Figure 4.

We also made experiments with other weighting functions. For example, we used a sigmoid function with two parameters, gain g and cutoff c:

$$w_{ij} \propto \frac{1}{1 + e^{g(c - z_{ij})}}$$

where z_{ij} is the magnitude of intensity gradient. Parameter c is usually set to 0 and the gain g is similar to parameter β in the previous equation: the higher g, the higher is the barrier at intensity discontinuities (see Figure 5). Both parameters seem to be very easy to interpret by an inexperienced user. Note that results with the exponential function of Equation 1 and $\beta = 20$ (Figure 4, bottom right) are quite similar to those with the sigmoid and $g = 10$ (Figure 5, right).

3.3. Perspective Constraints

Perspective information is a key piece when recovering the structure of some scenes. Several works [LCZ99, CRZ00] take advantage of these constraints by computing, or requiring from the user, the position of the horizon or camera parameters, information which will be used to recover the whole scene structure. In this work, we focus on ground-plane user input although additional cues could be used.

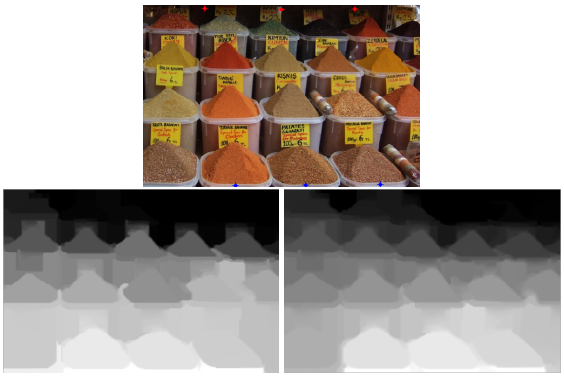

Figure 4: *Same original image than Figure 3 with 6 points selected (3 points in the far -red-, 3 in the near -blue-), and depths obtained with* $\beta = 10$ *(left) and* 20 *(right).*

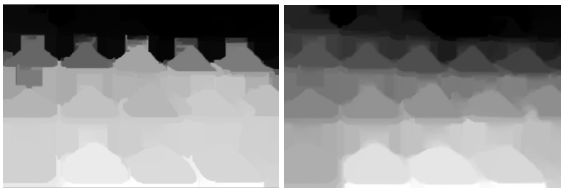

Figure 5: *Depths obtained with sigmoid weighting function,* $g = 5$ *(left) and* 10 *(right), for the original image and user input in Figure 4. Results for* $g = 5$ *are very similar to those of* $\beta = 20$ *in Figure 4.*

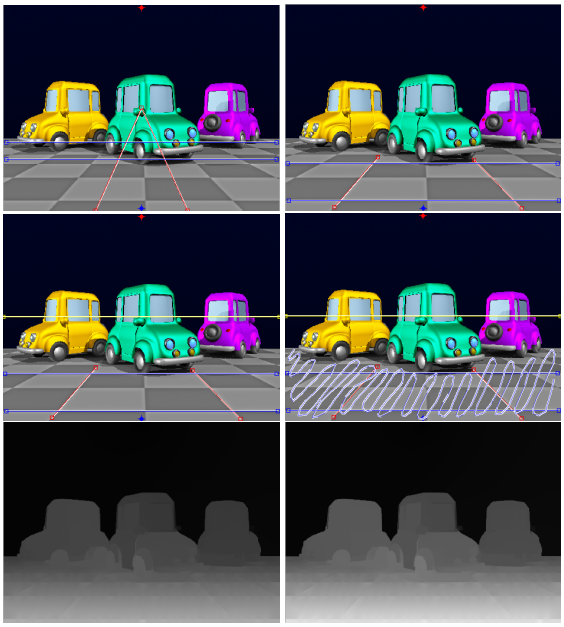

Figure 6: *Ground plane depth estimation: the user is given a set of straight lines (top-left) to be relocated (top-right), then the horizon line is computed (center-left, horizon in yellow), which together with a scribble (center-right, scribble in white-blue) is used to calculate the ground plane and the depths at the same time. Bottom: The resulting depth obtained with this user input (bottom-left) and with two more seed points added in the near (bottom-right).*

The user can draw a number of strokes to be considered as pixels that belong to the ground, as well as the position of the horizon. Although the user could indicate the horizon line directly, that is often a difficult task for untrained users. Therefore, we provide a simple interface to compute the horizon line by placing four straight lines. Figure 6 illustrates how the horizon line can be computed and how the ground pixels are assigned to a common ground plane. The user is given a set of straight lines (top-left) to be relocated (top-right) such that the paralelism and orthogonality properties can be used to obtain the horizon line (center-left). Parallel lines on the world intersect at vanishing points in the image, which lie on the vanishing line. Note that the vanishing line can be determined by means of other constraints [LCZ99] but we selected this one for its simplicity. The user places these lines in the ground (two red, two blue), such that the lines in identical color are parallel, while they are orthogonal to the lines in the other color.

The user can also draw a scribble to define the ground pixels (center-right in Figure 6). Let G be the set of points in the scribble and G_w the 3D ground points whose perspective projection correspond to the points in G. All the points $P_i = (x_i, y_i, z_i)^T$ in the ground are constrained to belong to the same 3D plane:

$$a_1 x_i + a_2 y_i + a_3 z_i + a_4 = 0, \quad \forall P_i \in G_w \tag{5}$$

where (a_1, a_2, a_3, a_4) are the plane parameters.

The perspective projection of any point P_i in the world into an image pixel $p_i = (u_i, v_i)^T$ can be modeled as

$$\begin{bmatrix} su_i \\ sv_i \\ s \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 & 0 \\ 0 & f & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} \tag{6}$$

where f is the focal length, $(u_0, v_0)^T$ is the principal point (the intersection between the image and the view direction) and s is a scale factor. This can be abbreviated as

$$s\tilde{p}_i = \begin{bmatrix} A & 0 \end{bmatrix} \tilde{P}_i \tag{7}$$

where $\tilde{p}_i$ and $\tilde{P}_i$ correspond to p_i and P_i in homogeneous coordinates and A is a 3×3 matrix that contains the camera parameters. This simplified camera model assumes that the physical angle between the u and v axes is $\frac{\pi}{2}$ and the aspect ratio of the camera is equal to 1 (the scales of u and v are identical). It is possible to compute the internal parameters

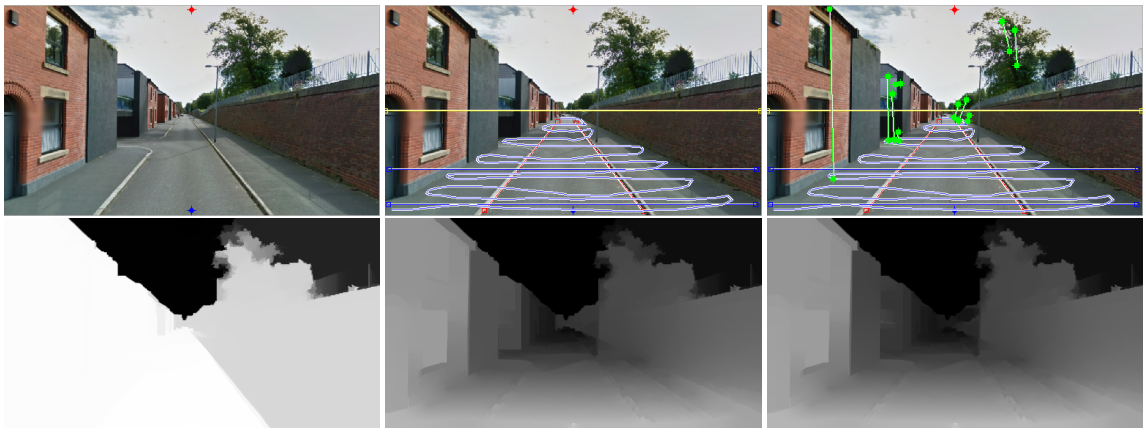

Figure 7: *Example of progressive addition of user input. Top row: image with input superimposed. Bottom row: depths corresponding to each input. From left to right: two seed points, addition of horizon and ground plane and addition of equality constraints.*

of the general camera calibration matrix from three vanishing points in three orthogonal directions [LCZ99]. Instead of that, we seek for an approximated solution and we further simplify the model by assuming the principal point to be at the center of the image. This is not always true, for example, if the image was cropped.

From the camera parameters A and the depth of a pixel z_i, we can recover the 3D coordinates of the point P_i by

$$\tilde{P}_i = \begin{bmatrix} z_i A^{-1} \tilde{p} \\ 1 \end{bmatrix} \tag{8}$$

where

$$A^{-1} = \begin{bmatrix} \frac{1}{f} & 0 & -\frac{u_0}{f} \\ 0 & \frac{1}{f} & \frac{-v_0}{f} \\ 0 & 0 & 1 \end{bmatrix}.$$

Equation 8 becomes

$$\tilde{P}_i = \begin{bmatrix} z_i(u_i - u_0)/f \\ z_i(v_i - v_0)/f \\ z_i \\ 1 \end{bmatrix} \tag{9}$$

As all the pixels in G_w fulfill Equation 5,

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \end{bmatrix} \tilde{P}_i = 0, \forall P_i \in G_w \tag{10}$$

From this equation we obtain

$$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \begin{bmatrix} (u_i - u_0)/f \\ (v_i - v_0)/f \\ 1 \end{bmatrix} = \frac{-a_4}{z_i} \tag{11}$$

Note that, if depths z_i and f were known, parameters a_i would be the only unknowns in this equation, and $-a_4$ acts a scale factor. As many works, we encode the estimated depths d_i as inversely proportional to z_i, such that $z_i = \infty$ corresponds to $d_i = 0$ and the minimum z_i corresponds to the maximum d_i. These d_i are similar to disparities in the area of stereo vision, except for true disparities the appropiate scale must be computed. Also, as we do not aim to recover the plane parameters, but only use them as a constraint on image depths, we can rewrite the three unknowns to include the scale factors so that the equation becomes

$$\begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix} \begin{bmatrix} (u_i - u_0)/f \\ (v_i - v_0)/f \\ 1 \end{bmatrix} = d_i \tag{12}$$

Finally, if f is known –many digital images include the focal length in their metadata–, parameters b_i are the unknowns in this equation, together with the depths, encoded as d_i. Let us define q_i as the pixel $p_i = (u_i, v_i)^T$ translated with respect to the image center, and divided by the focal length:

$$q_i = \begin{bmatrix} (u_i - u_0)/f \\ (v_i - v_0)/f \end{bmatrix} \tag{13}$$

Then, we can represent Equation 12 as:

$$B^T \tilde{q}_i = d_i$$

where $B = (b_1, b_2, b_3)^T$ is the set of unknowns that model the ground plane, and $\tilde{q}_i$ represents q_i in homogeneous coordinates.

This constraint is included into the optimization process as three additional unknowns that represent the 3D plane constraint, by forcing all the points in the scribble to follow it:

$$B^T \tilde{q}_g = d_g, \quad \forall p_g \in G$$

The points in the horizon (not the pixels themselves, but

the 3D virtual straight line) are also ground points that fulfill this constraint, which in addition are further constrained to be at the maximum depth ($d_i = 0$). Let H be a set of points in the horizon (in the experiments, we used only two). For any pixel $p_h \in H$, we can define a constraint:

$$B^T \tilde{q}_h = 0$$

where $\tilde{q}_h$ represents q_h in homogeneous coordinates, defined by Equation 13 from p_h.

Therefore, the optimization process becomes:

$$\begin{aligned} \text{minimize:} \quad & \sum_{\forall (p_i,p_j) \in E} w_{ij}(d_i - d_j)^2 \\ \text{subject to:} \quad & d_k = \hat{d}_k, \quad \forall p_k \in S \\ & B^T \tilde{q}_h = 0, \quad \forall p_h \in H \\ & B^T \tilde{q}_g = d_g, \quad \forall p_g \in G \end{aligned} \tag{14}$$

The resulting depths are shown in the bottom-left of Figure 6. The bottom-right depths are computed with the addition of two more points in the near part of the image. Let us note the differences between depths in Figures 6 and 2, which show results from the same example, with and without considering ground information, respectively. In Figure 2 the ground depth forms a kind of steps, due to prevention of propagation through high gradients. Therefore, depth changes occur between the tiles in the ground, and produce depth discontinuities, mainly horizontal steps. Ground depth in Figure 6 is much softer. The pixels in the scribble are assigned depths following the constraint, and these values are propagated to the rest of their neighbouring pixels. These neighbours still tend to form some irregularities in presence of high gradients, but in general depths vary more softly thanks to the ground plane constraint.

Figure 7 shows depths calculated from a view of a Manchester street obtained from the Google Street View, both without (left) and with (center) information about the horizon and the ground plane. The depths in the sides of the street are assigned depths somehow coherent with the ground depths. The examples in Figures 6, 7 and 12 were computed with the default value $f = 1$.

3.4. Equality and Inequality constraints

We allow the user to include an additional set of constraints in the form of equalities $Q_=$ and inequalities $Q_<$. In both cases, these constraints are given by the user through pairs of points. Let $Q_=$ be the set of pairs of pixels selected by the user to be at equal depth. We add:

$$d_a = d_b, \quad \forall (p_a, p_b) \in Q_= \tag{15}$$

into the list of constraints of Equation 14.

Figure 7 (right) shows the addition of ten equality constraints to an urban image. Figure 8 shows the addition of nineteen of these constraints to the example in Figure 6.

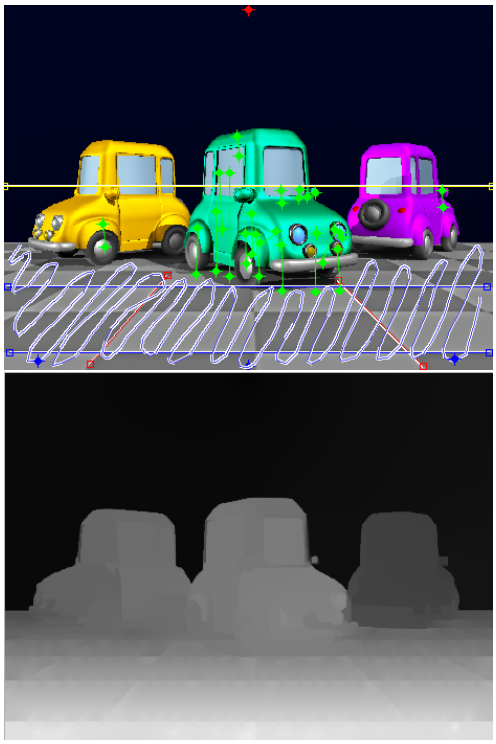

Figure 8: *Equality constraints: depth map resulting from the addition of nineteen pairs of pixels (linked in green) with equal depth to the example in Figure 6.*

Some pairs of pixels with equal depth were used to homogenize depths of the cars with their own wheels, other pairs were dedicated to the rear mirrors and the rest of pairs related the car in the middle to appropriate ground pixels to obtain a better depth ordering of the cars.

We can formulate as constraints not only equalities but also inequalities in pixel depths. The inequality constraint prevent regions of the image from having the same depth. The input is also in the form of pairs of points, and is added to Equation 14 as follows:

$$d_a - d_b < D, \quad \forall (p_a, p_b) \in Q_< \tag{16}$$

where $Q_<$ the set of pairs of pixels selected by the user, and D is the desired depth difference, which can be calculated as a percentage of the range of depth values, being in our experiments $D = 20$ for a depth range of 256.

Figure 9, right, shows an example of use of these inequality constraints, where six pairs of pixels with different depths are selected. The green point is constrained to be further than the yellow point of each pair. These points allow to refine depths in some areas of the face.

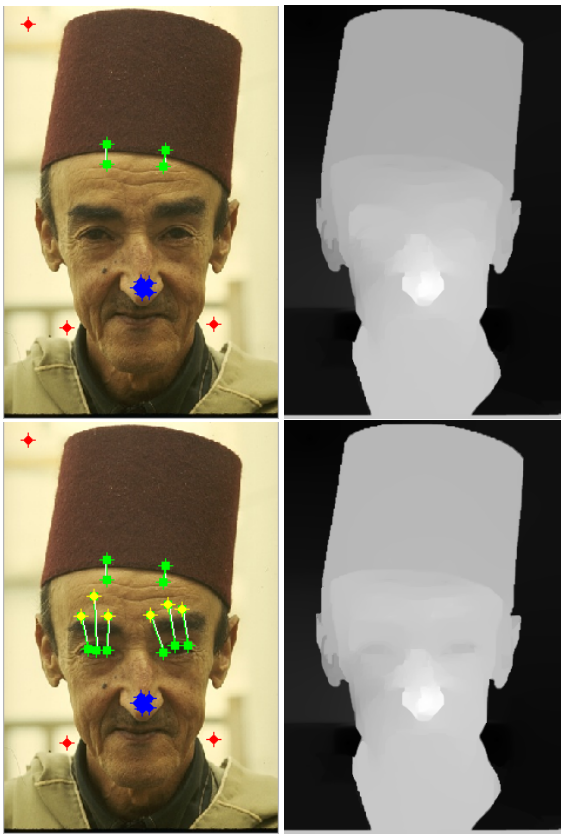

Figure 9: *Image from the Berkeley dataset [MFTM01] with input data superimposed (left) and the resulting depth (right). The addition of six pairs of inequalities (bottom row) allows to refine the resulting depth.*

4. Results

We use Matlab and the `cvx` library [BV04] to perform the convex optimization. Processing a standard image takes from seconds to tens of seconds, depending on the image size and the number of constraints. For example, the minimization for the 481×321 example in Figure 11 (first columnn), takes ten seconds, while the right example of Figure 10, with identical size, takes 30 seconds. The example in Figure 12, with size 1080×720, takes 130 seconds.

For the experiments we tested images from different sources. We used synthetic images like the example in Figure 2. We experimented with real images from the Berkeley segmentation dataset [MFTM01], like the ones in Figures 9-10. We also considered the special case of urban images, by using Google street view (Figures 7 and 12).

For some scenes, the amount of user input required is minimal, like the first column of Figure 11, while others require much more user input (see Figure 10). In other scenes, which apparently seem more difficult, we obtain quite satisfactory results with a few hints (see second and third columns of Figure 11). Although in several cases it is not easy to preview the optimal combination of hints, the user can add hints progressively (like the examples in Figures 7 and 9, to succesively refine the depth map. Moreover, many of the hints consist on barely one or two clicks (i.e. Figure 11), which is easy to learn by the user.

As opposed to some automatic approaches [SSN09, LGK10, HB13] which require accurate depth maps and performs quantitative comparisons with the ground truth; results from methods that rely on user interaction are more difficult to measure. Such depth maps can be refined progressively to match a desired output, which frequently do not need to be identical to the reality. Comparison should be performed under identical conditions. For example, [YSHSH13] used the same user input data than [WLF*11] in order to allow acurate comparison of their results, but our user input data is different from this. As our method also relies on the user perception and interaction and obtains relative depth values, we provide the resulting depth maps for qualitative evaluation and we use images from the Berkeley database, among others.

5. Conclusions

We have developed a method to obtain an approximated depth map from a single image which relies on the user ability to interpret 2D images. We allow two types of constraints provided by the user: perspective constraints and equality/inequality constraints. These constraints are incorporated into an optimization framework, whose minimal input required is one point in the farthest and one point in the nearest regions of the image. Depth values are propagated throughout the scene leveraging gradient information.

There are several ways to improve this work. For example, due to the flexibility of the optimization framework, it would be possible to incorporate additional constraints derived from automatic methods such as blur produced by defocus [EL93], or low-level depth ordering features [CC13]. Current input could be transformed to scribbles [YSHSH13] to minimize interaction. New tools could be developed to improve accuracy, such as additional hints on plane orientation or basic modeling tools to establish geometrical constraints. Finally, although the processing times are still high to be suitable for interaction, we also plan to check other optimization tools to reduce the computational effort.

Acknowledgements

We want to thank the reviewers for their helpful comments, Jose Ribelles for insightful discussions, Carlos Aliaga for the synthetic image of cars and Yotam Gingold for sharing his input images. This work was supported by the European Commission, Seventh Framework Programme, through projects GOLEM (Marie Curie IAPP, grant: 251415) and

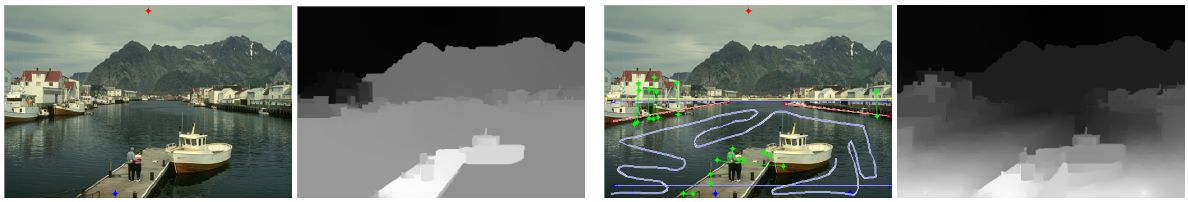

Figure 10: *Left pair, image from the Berkeley dataset [MFTM01] with only two seeds (near and far). Right pair, resulting depth map adding the equality and perspective constrains shown in the picture.*

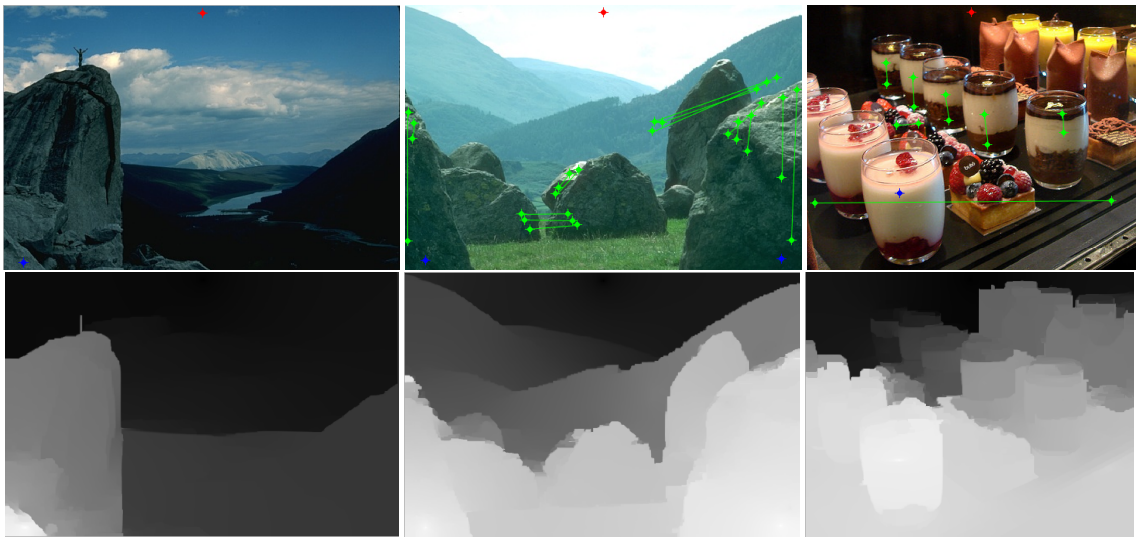

Figure 11: *Two images from the Berkeley dataset [MFTM01] (left) and one image courtesy of Yotam Gingold (right), with user input superimposed (top row), and the obtained depths (bottom row).*

VERVE (ICT, grant: 288914), the Spanish Ministry of Science and Technology (TIN2010-21543). The Gobierno de Aragón additionally provided support through the TAMA project and a grant to Elena Garces.

References

[ART10] AMER M., RAICH R., TODOROVIC S.: Monocular Extraction of 2.1D Sketch. In *ICIP* (2010). 2

[BV04] BOYD S., VANDENBERGHE L.: *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004. 8

[CC13] CALDERERO F., CASELLES V.: Recovering Relative Depth from Low-Level Features Without Explicit T-junction Detection and Interpretation. *International Journal of Computer Vision* (Feb. 2013). 1, 2, 8

[CCM96] CASELLES V., COLL B., MOREL J.-M.: A kanizsa programme. In *Variational Methods for Discontinuous Structures*. Springer, 1996, pp. 35–55. 2

[CRZ00] CRIMINISI A., REID I., ZISSERMAN A.: Single View Metrology. *IJCV 40*, 2 (2000), 123–148. 2, 4

[EL93] ENS J., LAWRENCE P.: An investigation of methods for determining depth from focus. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 15*, 2 (1993), 97–108. 8

[GIZ09] GINGOLD Y., IGARASHI T., ZORIN D.: Structured annotations for 2D-to-3D modeling. *ACM Transactions on Graphics (TOG) 28*, 5 (2009), 148. 1, 3

[Gra06] GRADY L.: Random walks for image segmentation. *IEEE transactions on pattern analysis and machine intelligence 28*, 11 (2006), 1768–1783. 3

[GSCO12] GINGOLD Y., SHAMIR A., COHEN-OR D.: Micro perceptual human computation. *ACM Transactions on Graphics (TOG) 31*, 5 (Aug. 2012), 119:1–119:12. doi:10.1145/2231816.2231817. 2

[GWZS07] GAO R.-X., WU T.-F., ZHU S.-C., SANG N.: Bayesian inference for layer representation with mixed markov random field. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Yuille A., Zhu S.-C., Cremers D., Wang Y., (Eds.), vol. 4679 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2007, pp. 213–224. 2

[HB13] HASSNER T., BASRI R.: Single view depth estimation from examples. *CoRR abs/1304.3915* (2013). 1, 2, 8

[HC12] HAINES O., CALWAY A.: Detecting planes and estimating their orientation from a single image. In *Proceedings of the British Machine Vision Conference* (2012), BMVA Press, pp. 31.1–31.11. 2

[HEH11] HOIEM D., EFROS A. A., HEBERT M.: Recovering Occlusion Boundaries from an Image. *IJCV 91*, 3 (2011). 1, 2

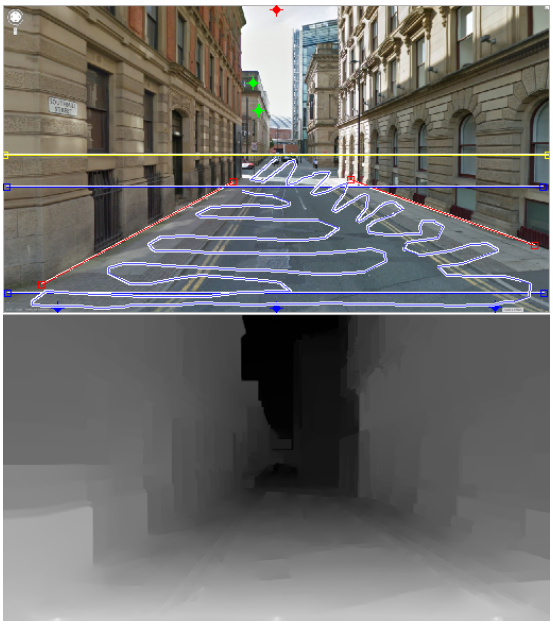

Figure 12: *Top, image of a Manchester street from the Google Street View, with user input superimposed. Bottom, depths obtained from optimization.*

[KZP*13] KIM C., ZIMMER H., PRITCH Y., SORKINE-HORNUNG A., GROSS M.: Scene reconstruction from high spatio-angular resolution light fields. *ACM Trans. Graph. 32*, 4 (July 2013), 73:1–73:12. 1

[LCZ99] LIEBOWITZ D., CRIMINISI A., ZISSERMAN A.: Creating Architectural Models from Images. *Computer Graphics Forum 18*, 3 (1999), 39–50. 2, 4, 5, 6

[LGK10] LIU B., GOULD S., KOLLER D.: Single Image Depth Estimation From Predicted Semantic Labels. In *CVPR* (2010), pp. 1253–1260. 2, 8

[LHK09] LEE D., HEBERT M., KANADE T.: Geometric reasoning for single image structure recovery. In *CVPR* (June 2009), Ieee, pp. 2136–2143. 2

[Mai10] MAIRE M.: Simultaneous Segmentation and Figure/Ground Organization Using Angular Embedding. *Lecture Notes in Computer Science 6312* (2010), 450–464. 2

[MFTM01] MARTIN D., FOWLKES C., TAL D., MALIK J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision* (July 2001), vol. 2, pp. 416–423. 8, 9

[NM90] NITZBERG M., MUMFORD D.: The 2.1-D Sketch. In *ICCV* (1990), pp. 138–144. 2

[OCDD01] OH B. M., CHEN M., DORSEY J., DURAND F.: Image-based modeling and photo editing. *ACM SIGGRAPH* (2001), 433–442. 1, 2

[PS13] PALOU G., SALEMBIER P.: Monocular depth ordering using t-junctions and convexity occlusion cues. *IEEE Transactions on Image Processing 22*, 5 (2013), 1926–1939. 2

[PZF06] PRASAD M., ZISSERMAN A., FITZGIBBON A.: Single view reconstruction of curved surfaces. *CVPR* (2006). 2

[RB13] RAMALINGAM S., BRAND M.: Lifting 3d manhattan lines from a single image. In *ICCV* (2013), IEEE, pp. 497–504. 2

[SCD*06] SEITZ S. M., CURLESS B., DIEBEL J., SCHARSTEIN D., SZELISKI R.: A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1* (Washington, DC, USA, 2006), CVPR '06, IEEE Computer Society, pp. 519–528. 1

[SSJ*10] SÝKORA D., SEDLACEK D., JINCHAO S., DINGLIANA J., COLLINS S.: Adding Depth to Cartoons Using Sparse Depth (In) equalities. *Computer Graphics Forum 29*, 2 (2010), 615–623. 3

[SSN09] SAXENA A., SUN M., NG A. Y.: Make3D: learning 3D scene structure from a single still image. *PAMI 31*, 5 (2009), 824–840. 1, 2, 8

[WLF*11] WANG O., LANG M., FREI M., HORNUNG A., SMOLIC A., GROSS M.: Stereobrush: interactive 2d to 3d conversion using discontinuous warps. In *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling* (New York, NY, USA, 2011), SBIM '11, ACM, pp. 47–54. 1, 3, 8

[YJHS12] YÜCER K., JACOBSON A., HORNUNG A., SORKINE O.: Transfusive image manipulation. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA) 31*, 6 (2012), 176:1–176:9. 3

[YSHSH13] YÜCER K., SORKINE-HORNUNG A., SORKINE-HORNUNG O.: Transfusive weights for content-aware image manipulation. In *Proceedings of the Vision, Modeling and Visualization Workshop (VMV)* (2013), Eurographics Association. 1, 3, 8