Pylon Grid: A Fast Method for Human Head Detection in Range Images

Rusi Antonov Filipov

Faculty of Computer Science

Hochschule Karlsruhe

Flávio Luis Cardeal Pádua, Marco Aurélio Buono Carone

Department of Computing

Federal Center of Technological Education of Minas Gerais

Abstract

We present a fast and accurate method for human head detection in range images captured by a stereo camera that is positioned vertically, pointing from the roof to the ground. We show how a static grid of measure points (pylons) can outperform hill climbing techniques and how a fast median filter can be used for effective preprocessing of the range data. The Pylon Grid algorithm detects all local minima in the range image and has a linear time complexity in respect to the number of pylons. One important prerequisite for applying the Pylon Grid algorithm to human head detection is a one-to-one relationship between human heads in the scene and local minima in the range image. This is achieved in a preprocessing phase, where an ortographic projection, convexization and noise filtering is applied to the range data. The preprocessing steps also run in linear time and can be parametrized for a further trade-off between computational cost and accuracy. The method was tested with crowded scenes, where multiple dense groups of up to six people move in random directions and have physical contact.

Keywords: stereoscopy, range image, head detection, local minimum search, hill climbing, perspective distortion, orthographic projection, structural convexization, computer vision, people counting

Preprint submitted to NEUROCOMPUTING Special Issue on Behaviours in VideoNovember 5, 2011

Figure 1: People flow at a shopping mall, a subway and an airport

Figure 2: A dense scene (center) captured from above and two corresponding range images. The students heads appear at the darkest spots in the blobs. The right image was extracted using *sub-pixel interpolation* and has higher range resolution.

1. Introduction

There is a class of applications, that make use of automatic tracking and counting of people in a scene. One example is consumer path tracking in shopping malls in order to collect statistical data about the consumers behaviour. Another application people counting and pedestrian flow analysis in public areas.

An essential component of any tracking system is the *detector*, which determines the positions of the people in each frame of the image sequence. In the context of stereo vision, the problem of precise detection can be attacked effectively with a vertical camera orientation (Figure 1).

While the vertical camera orientation limits the size of the observed area, it provides a clear view, with minimal occlusions among the people. This enables a reliable segmentation of the people from the ground, with accurate head detection, even in crowded scenes with hight people density.

Another advantage of the vertical camera orientation is that human heads appear as local minima in the range image (Figure 2). Considering this, and

assuming that human bodies have convex silhouettes in the range image, the problem of head detection can be reduced to finding all local minima in the range image.

2. Related Work

Vision-based algorithms for the people counting and tracking problem may be divided into two main classes: the *direct* and the *indirect* methods [1]. The direct methods [2, 3, 4, 5] are based on the detection of each person in the scene, by using some form of segmentation and object detection algorithm. The counting, in turn, is performed as a second step. The indirect methods [1, 6, 7, 8] instead, perform the people counting by using the measurement of some features that do not require the separate detection of each person in the environment.

This work presents a new algorithm, the Pylon Grid, which is used to detect efficiently human heads in range images. This algorithm lies at the heart of a novel direct vision-based approach to estimate people flow in both indoor and outdoor environments.

In [1], the authors focus on large groups of people which partially occlude each other. They propose a clustering algorithm that estimates the point density in each cluster and the distance to the camera, which faces the scene from an angle of about 45°. Khan and Shah [2] propose a multi-view approach to solving the problem of occlusion and lack of visibility in crowded and cluttered scenes. They gather evidence from every camera and use purely image-based 2D constructs. The authors of [3] attack the problem of occlusion with a Bayesian framework. In [4] the authors employ a learning approach that builds models of the people while the tracking. The authors of [5] focus on estimating the number of people in a big crowded scene.

In our method, we focus on small scenes instead and attack the occlusion problem with a vertical camera orientation. Our detector is designed to precisely detect each individual, because accurate detection can be a solid base for robust tracking and counting.

3. Similar Schemes

3.1. Hill Climbing

There is a class of *hill-climbing* algorithms that are able to find one local minimum in sublinear time. The approach is based on putting a climber

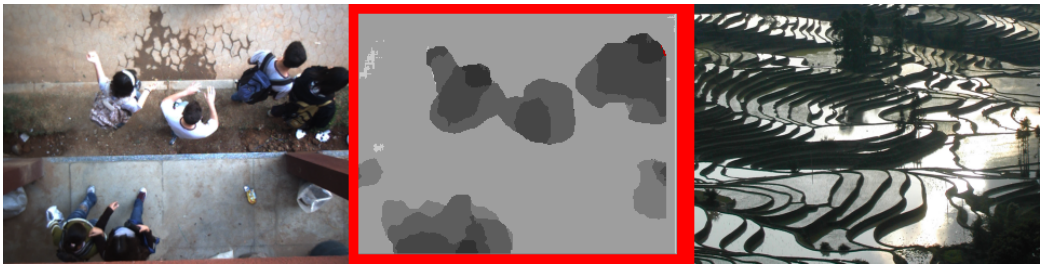

Figure 3: The range image contains plateaus that resemble a rice field.

somewhere in the range image and then check a the surrounding neighbor pixels at a close distance. The climber descends by moving to the neighbor pixel with the smallest value. He and continues the search from there, until he reaches a position where no neighbour has a smaller value, which is a local minimum.

This method is well suited for strictly monotonous functions, with local minima that spread over just a few positions. Unfortunately, one inherent property of range images is that they are *discrete functions*, where each pixel can only have a certain discrete value. This effect is visible on Figure 3 (center), where the student's heads are covered by multiple pixels with the same range value. We can also observe that some areas of the upper body (shoulders and backpacks) have the same range value. Once a climber reaches such a *plateau*, it is hard to decide in which direction to continue the search.

The discrete values in the range image originate from the *disparity image* captured by the stereo camera, which is also stores only discrete values which represent the correlation of a pixel in the left image to the same pixel in the right image. Moreover, the range resolution of the range image decreases when the camera is placed higher above the ground (Figure 4).

Figure 5 shows some difficult situations for in a hill climbing approach. In (a) it is difficult to decide where to start the search with the climber and in (d) it is hard to chose in which direction to continue, once a plateau is reached. In (c) the climber must decide where to continue his path, if two possible moves are equally adequate. Furthermore, a strict requirement of the problem presented in this text is to find the positions of *all* local minima in the range image, not just any local minimum. This is difficult to achieve with a single climber. And without knowing the number of human heads beforehand, it is hard to decide how many climbers to use (5 (b)).

Yet another goal is to find the *center* of the head plateau. In (e) the

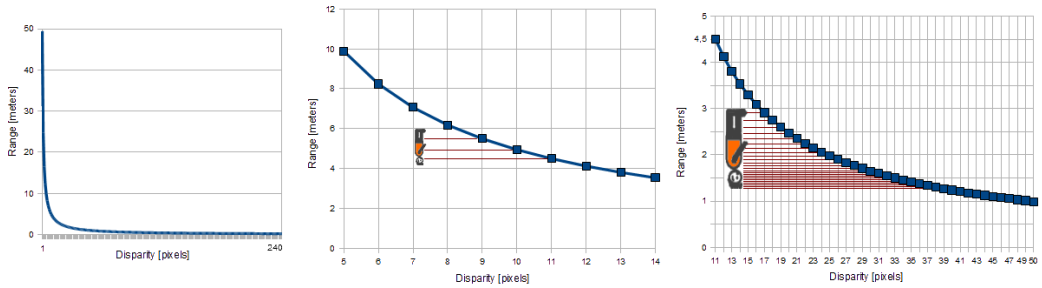

Figure 4: discrete disparity to range function of the Bumblebee2 stereo camera. The higher the camera is placed above the ground, the fewer range levels has the range image.

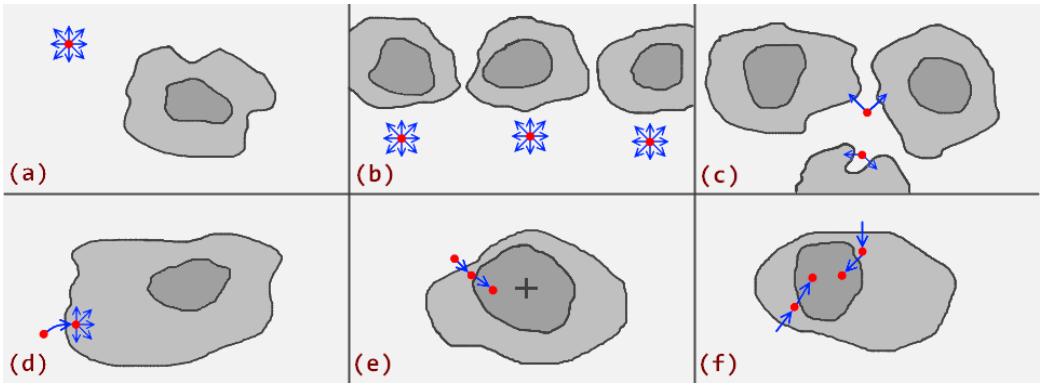

Figure 5: Problematic situations in a hill climbing approach.

climber has reached the minimum, but now the problem is to find its center. Situation (f) shows two climbers that found the same minimum this requires special care in order not to count the minimum twice.

3.2. Watershed

The Watershed algorithm gradually "fills" layers of the range image with imaginary water, until all regions are filled enough to touch each other, so that borders between the filled regions emerge. As a result, the whole scene is divided into multiple adjacent cells. Hoang and Won use a marker-free Watershed algorithm for segmentation of protein spots [9]. Rambabu and Chakrabarti propose a combined Watershed and Hillclimbing technique targeting a hardware implementation [10]. While the Watershed method places a cell around every local minimum, the geometry of the cell does not accurately reflect the shape of the human silhouette, and the position of the

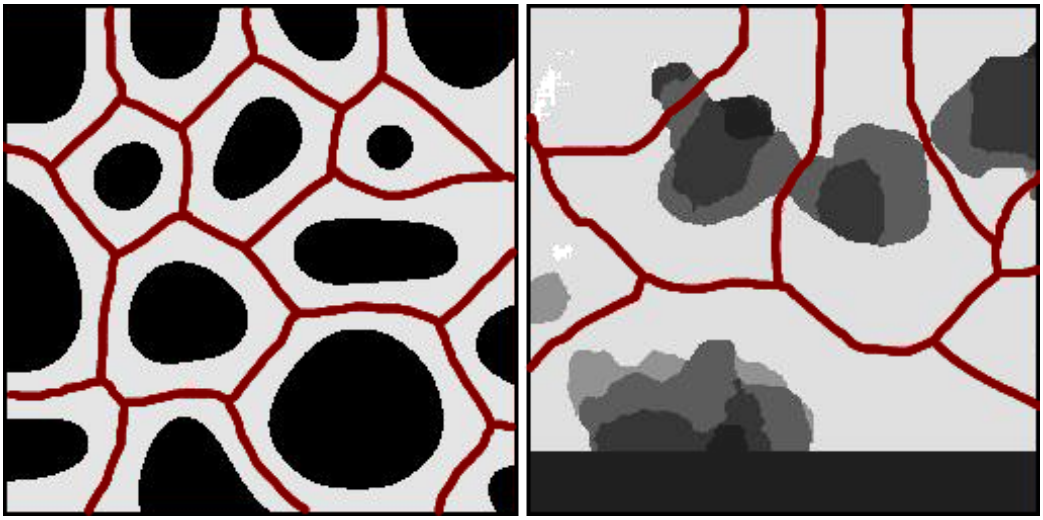

Figure 6: Watershed applied to a sample image (left) and to a realistic range image (right).

local minimum within the cell is not obvious. This happens because the cell's shape depends on the relative positions of the adjacent local minima. This makes it hard find the head's center in the cell reliably, especially in scenes with sparse or non-uniform distribution of the people (Figure 6).

3.3. Contour Extraction

In this approach the body contours of the people are extracted from the range image. The contours are extracted by taking advantage of the fact that points that belong to the body show smooth changes in their distance values. At object borders, discontinuities emerge that cause an edge in the range image that in turn is identified by segmentation algorithms [11]. The authors binarize the range image using adaptive thresholding and then apply a morphological opening to extract the contours. Having the contour image, they apply shape classification to recognize which contours belong to human bodies.

3.4. Model Fitting

Another approach is model-based fitting, where the range data is segmented and then the body regions are fit into a 3-D model which describes the shape of human body and head. A classifier determines if the particular blob fits well into the template and eventually classifies it as as a head. The

authors of [12] propose a robust ellipsoidal model fitting of human heads. In our method we don't use a special 3-D model, nor any fitting. Instead, we assume that all local minima will be located at the tops of the human heads, and convexize the data in a preprocessing step so that this assumption is satisfied.

4. Methodology

First we will discuss the ideal case where the one-to-one relationship between local minima and human heads is given, and human bodies have a convex range silhouette with one minimum at the top of the head.

4.1. Segmentation of the People

One advantage of a stereo over 2-D image processing is the extremely robust segmentation of the people from the ground [13]. We assume that human heads will only be located at a certain [min..max] valid range. Range values below or above these thresholds are considered either irrelevant, invalid or noise. With the vertical camera orientation the stereo-based segmentation becomes merely a comparison of the range value to two thresholds.

4.2. Stereo Model for Humans

As we said in the introduction, the problem of finding all human heads in the scene can be reduced to the problem of finding all local minima in the range image, but only if there is a one-to-one relationship between human heads in the scene and local minima in the range image. This assumption means that:

- For every human head in the scene there is exactly one local minimum that appears in the range image.
- For every local minimum that appears in the range image there is exactly one human head in the scene.

In an ideal range image, the one-to-one relationship is always given. But as we can see in Figure 7, the head of the boy does not have a clear local minimum, and his hand creates one instead. Even in range images with quite good range data and no people rising hands, we usually observe more local minima than human heads due to natural shapes like shoulders, and also due to noise artifacts. Later we will discuss a strategy of preprocessing the data, where the one-to-one assumption is enforced by a *structural convexization* of the range data.

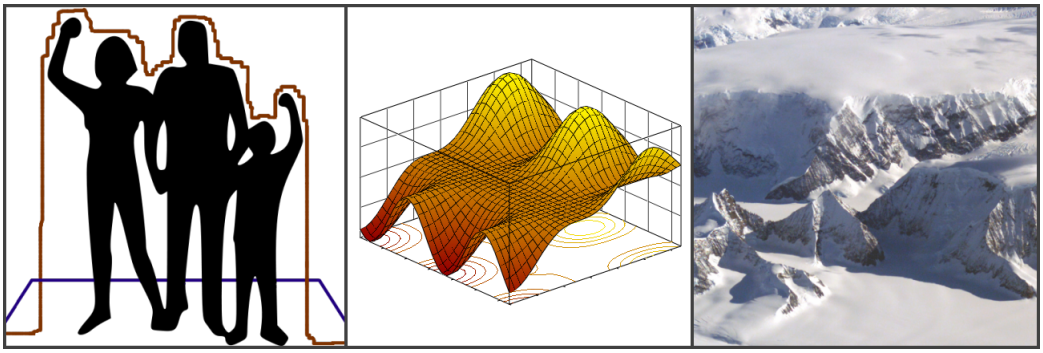

Figure 7: Three mental models for the humans in a range image.

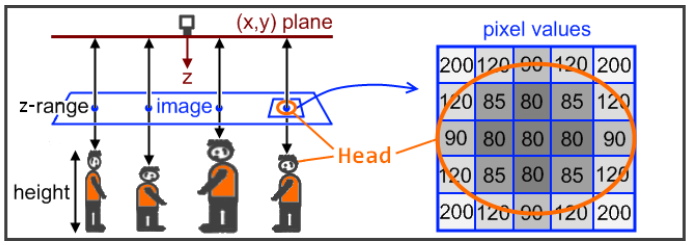

Figure 8: Vertical ranges (z-ranges) encoded in the pixel values of the range image.

4.3. The Range Data

Each pixel in the range image stores a value that corresponds to the vertical range (z-range) from the camera plane to the object that is covered by the pixel (Figure 8). We used 8-bit greyscale images, where values in the interval [0..255] were mapped to actual ranges in meters.

The proposed algorithm operates solely on the range data of a single image, and doesn't use any color information. The detector doesn't use any knowledge from the analysis of previous images and it doesn't learn over time.

4.4. The Pylon Grid Algorithm

As we showed earlier, hill climbing algorithms face various problems of *choice* in the context of a discrete function – where to start the search, how many climbers to use, how to proceed in plateau areas, how to handle ambigous situations, how not to count a minimum twice. The Pylon Grid algorithm eliminates all these problems by using a *static* grid of measure points – pylons – instead of moving climbers.

Figure 9: The Pylon Grid (pylons are the white points) is applied to the range image on the right.

The pylons are distributed equidistantly in a sparse grid over the range image (Figure 9). Every n-th pixel is covered by a pylon, where n the pixel-distance between the pylons. The pylon stores the range value of its pixel and its pixel coordinates. Every pylon is connected to his four direct neighbours, so that adjacent pylons can be compared by range and the whole grid can be traversed.

The distance between the pylons is configurable and depends on the camera height and its focal angle. The higher the camera is positioned, the smaller the people appear in the range image and the denser the grid needs to be, in order to cover every human with a pylon that is close to the center of the head.

The input of the algorithm is a range image. The output is a list that contains the (x,y) pixel coordinates of the centers of the detected human heads. The Pylon data structure:

```
class Pylon
{
    int ImageX;      // x-coordinate (row) of the pylon's pixel
    int ImageY;      // y-coordinate (column) of the pylon's pixel
    byte Range;      // the range value of the pylon's pixel
    bool IsHead;     // marker flag which is true for head pylons
    bool IsGrouped;  // indicates if this pylon is grouped to a head group
}
```

The pylon objects are stored in a 2-D array (the grid), so that the array

indices can be used to traverse the grid and compare adjacent pylons. The pixel coordinates and the range are read-only. A mutable flag `IsHead` states if the pylon is a head pylon (placed on a local minimum area) or not.

Step 1 - Initialization of the grid. The 2-D array is filled with pylon objects which are intialized with the (x,y) coordinates and the range value of their pixel. The pylons at the borders of the image are always considered non-head and their `IsHead` flag is set to `false`. All the remaining inner pylons are initially assumed to be head pylons (even if that may be wrong) and their `IsHead` flag is set to `true`. The initialization of the pylons in the 2-D array is a linear time operation in respect to the number of pylons.

Pseudocode:

```
grid = new Pylon[gridRows][gridCols]
for rowIndex in grid:
  for colIndex in grid:
    Pixel pixel = getPixelForPylon(rowIndex, colIndex)
    Pylon pylon = new Pylon(pixel.X, pixel.Y, pixel.Value, true)
    grid[rowIndex][colIndex] = pylon
```

Step 2 - Discarding of the non-head pylons. In this phase the goal is to find and mark all the *non-head* pylons and set their `IsHead` flag to `false`. It is not possible to say if a given pylon is a head pylon just by comparing it to his direct four neighbours. But if pylon A has a direct neighbour B with a lower range value than A, then pylon A is definitely *not* a local minimum. In the case where A and B have equal ranges then we must examine the neighbors of B, and eventually their neighbours in order to be able to discard A. For this reason, the grid needs to be traversed *recursively* for every pylon that is not yet visited:

If the current pylon is already marked as non-head, it is not processed (this breaks the current recursion path). Otherwise, the current pylon is compared to each of his four adjacent neighbors and is marked as non-head if any of the following conditions is true:

- A neighbor pylon has a smaller value than the current pylon.
- A neighbor pylon has the same value, but the neighbor pylon is already marked as non-head.

If the current pylon gets marked as non-head, the traversal is propagated recursively to each of his neighbours that are still marked as head. Otherwise, the recursion is terminated. After all pylons from the grid have been

processed, only the head pylons remain unchanged with their `IsHead` flag still left to `true`. Pseudo Code:

```
for all pylons in the grid:

void DiscardTailPylon(int gridRow, int gridCol, Pylon previous)
{
    Pylon current = grid[gridRow, gridCol];
    if (!current.IsHead)
        return;

    Pylon adjacentE = grid[gridRow, gridCol + 1];
    Pylon adjacentW = grid[gridRow, gridCol - 1];
    Pylon adjacentN = grid[gridRow - 1, gridCol];
    Pylon adjacentS = grid[gridRow + 1, gridCol];

    bool visitedE = (previous == adjacentE) || current.IsDiscardedBy(adjacentE);
    bool visitedW = (previous == adjacentW) || current.IsDiscardedBy(adjacentW);
    bool visitedN = (previous == adjacentN) || current.IsDiscardedBy(adjacentN);
    bool visitedS = (previous == adjacentS) || current.IsDiscardedBy(adjacentS);

    if (visitedE || visitedN || visitedW || visitedS) {
        current.IsHead = false;
        if (!visitedE)  // go E
            DiscardTailPylon(gridRow, gridCol + 1, current);
        if (!visitedN)  // go N
            DiscardTailPylon(gridRow - 1, gridCol, current);
        if (!visitedW)  // go W
            DiscardTailPylon(gridRow, gridCol - 1, current);
        if (!visitedS)  // go S
            DiscardTailPylon(gridRow + 1, gridCol, current);
    }
}
```

A single recursive traversal, especially the first one, will usually mark many pylons, but not all of the pylons. Therefore the recursive search is launched for every pylon in the grid. Note that even though the recursive search is launched for every pylon in the grid, it terminates immediately for already discarded non-head pylons, and every head pylon is visited only once

(only non-head pylons propagate the recursion). As a result, this step is also linear in respect to the number of pylons.

Step 3 - Grouping of adjacent head pylons. Usually a small group of a few head pylons will cover the same head. The goal of the grouping phase is to determine which head pylons belong to which group. A group is a disjunct set of connected head pylons that are all on top of a local minimum plane.

First, a new empty group is created and the every python from the grid is examined. The pylon is added to the group if both conditions are true:

- The current pylon is marked as a head pylon.
- The current pylon has not been added to a group yet.

Pseudo Code:

```
for all pylons in the grid:

void GroupHeadPylon(int gridRow, int gridCol, ref PylonGroup pylonGroup)
{
Pylon pylon = grid[gridRow, gridCol];
    if (!pylon.IsHead || pylon.IsGrouped)
        return;

    pylonGroup.Register(pylon);

    GroupHeadPylon(gridRow - 1, gridCol, ref pylonGroup);
    GroupHeadPylon(gridRow + 1, gridCol, ref pylonGroup);
    GroupHeadPylon(gridRow, gridCol - 1, ref pylonGroup);
    GroupHeadPylon(gridRow, gridCol + 1, ref pylonGroup);
}
```

This step is also recursive if the pylon is added to the group, a recursive search to his neighbors is started so the adjacent head pylons can be added to the group too. At the end, the group contains all head pylons that cover together the same head. Now the next empty group is created, and the rest of the pylons are processed. This step is also linear in respect to the number of pylons, because only the ungrouped pylons are processed recursively.

Step 4 - Calculating the head centers. Having the groups of all head pylons, now only the heads center has to be computed. This is done by

computing the average (x,y) position of the pylons in the same group (the center of mass):

$$HeadCenter_x = \frac{\sum PylonImg_x}{GroupSize} \qquad HeadCenter_y = \frac{\sum PylonImg_y}{GroupSize} \quad (1)$$

This step is also linear in respect to the number of pylons.

4.5. Runtime Characteristics

Since all steps of the algorithm run in linear time in respect to the number of pylons, the whole algorithm also runs in linear time, because it consists of the four steps, applied sequentially. The algorithm is predictable and deterministic.

Because the Pylon Grid is static, the algorithm guarantees to find every head, to report every head only once, and do this in a guaranteed maximal time - something that is hard to achieve with hill-climbing approaches. The computational efficiency comes mainly from the fact that only the pylon pixels are examined.

5. Preprocessing of the Range Data

Even when the stereo parameters are well-calibrated to the scene, the range images can still contain invalid or incomplete data, so that the one-to-one relationship between human heads in the scene and local minima in the range image does not hold. We apply a number of preprocessing steps in order to enhance the range data so that that it fits into the assumption of the one-to-one relationship.

5.1. Handling "Mud Stains" Noise

The typical noise that we encountered was a kind of dark mud stains that appear frequently at the borders of the range silhouettes of the people (Figure 10).

The *median filter* turned out to be especially well suited for removing this kind of noise.

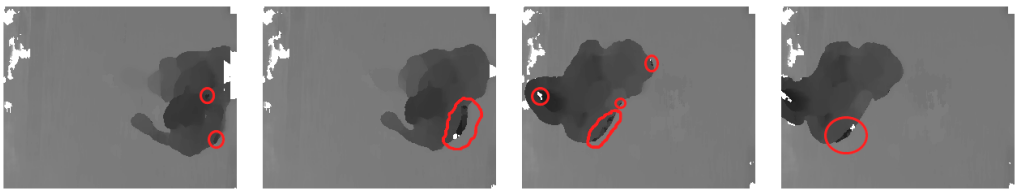

Figure 10: Noisy mud stains in a sequence of range images.

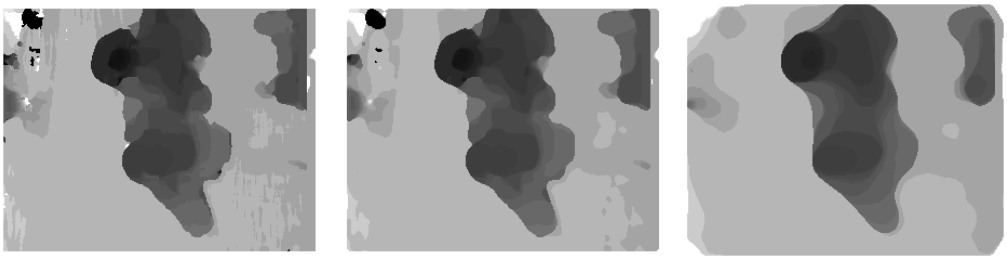

Figure 11: A range image (left) passed through a median filter with kernel radius 3 (middle) and kernel radius 15 (right).

5.2. A Fast Median Filter

From a qualitative viewpoint, the median filter performed very well for filtering the mud stains.

Unfortunately, while the noise removal ability of the median filter is excellent, it is also considered slow, because of its *non-linear* computational complexity. This comes from the need to pre-sort all values in order to pick the median value.

The fastest known general-purpose sorting algorithms have a $O(n \cdot ld(n))$ complexity. But the median filter can be made linear, by adopting the special-purpose *counting sort* algorithm, which is linear [14].

Counting sort is a specialized method for sorting integral values with linear time complexity. It exploits the fact that the range of all possible values is known. For example, if we sort byte values (which is very common in greyscale images), we know that all values will be in the range [0..255].

Now before the sorting starts an array of counters is initialized one counter for every possible value (a histogram). In the case of byte values it would be an array of 256 counters, and the counters index corresponds to the counted value (Figure 13).

Having initialized all counters to zero, the sorter goes sequentially through

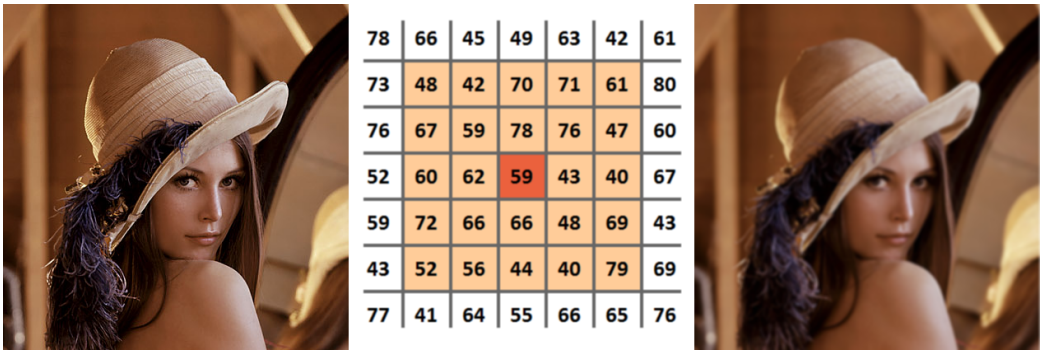

Figure 12: The Median filter in action.

Figure 13: This counter array is a histogram of the 8-bit grayscale values [0..255].

the unsorted input data (only once) and increments the corresponding counter for every input element it sees. Now the counter array represents a histogram of the input. In the final step the sorter goes linearly through the counter array (only once) and outputs the counters index (his corresponding element) as many times as the counters value. And the produced output is just the sorted input! Figure 14 shows an example of applying counting sort.

While counting sort makes the median filter linear, another performance gain comes from the fact that the median filter is applied only at the pylon pixels in the Pylon Grid (not on every pixel in the range image), because

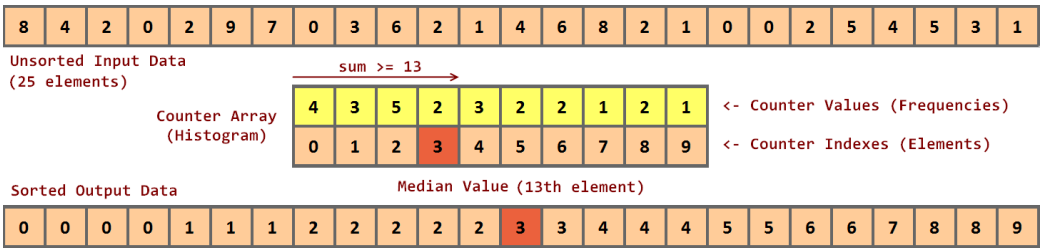

Figure 14: Counting sort applied to known range of possible values [0..9].

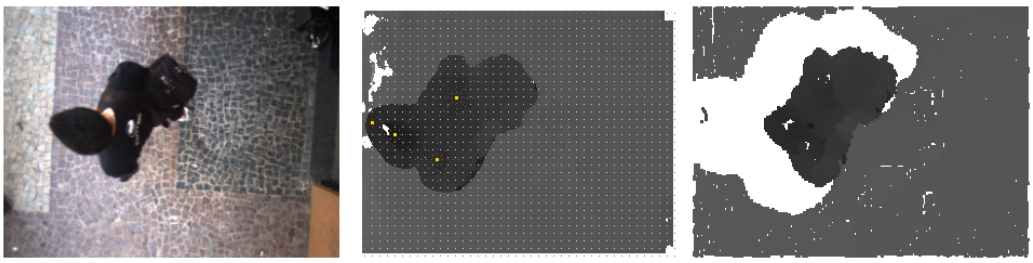

Figure 15: The range image in the middle contains one head and 3 local minima. The image on the right is an orthographic projection of the image in the middle, and has only one local minimum at the head.

only the pylon pixels are compared when the local minima are detected.

5.3. Handling Perspective Distortion

When the camera is installed low over the ground (2-3 meters), the perspective effect is clearly visible when people are not in the center of the image. This can be observed in Figure 15 (left) where the person is moving away from the image center to the image border.

The persons range silhouette (middle) contains to a big part the persons back. In fact, the back occupies about twice as much area as the head on the range silhouette. And this is a problem, because not only the persons head but also his shoulders appear as local minima. Another problem caused by the perspective effect is that the range silhouette of the same person looks compact in the middle of the scene and elongated near the image borders. Since the shape is changing, it is hard to find stable features that can be used for a robust head classification.

One way to eliminate the upper issues is to put the camera very high over the ground. But this is not always a convenient option and also lowers the range-resolution of the image. Another option is to transform the range image using an orthographic projection like shown on Figure 16.

The orthographic projection verticalises the range image by shifting the pixels from their original position along the imaginary line that passes through the image center and the pixel in question. The result looks as if we were looking at the image from many viewpoints simultaneously and straight vertically to the ground. Some objects that are vertically above each other in the scene will disappear in the orthographic view.

When this transformation is applied to the human silhouettes in the range image, only the head and the upper part of the chest and the shoulders remain

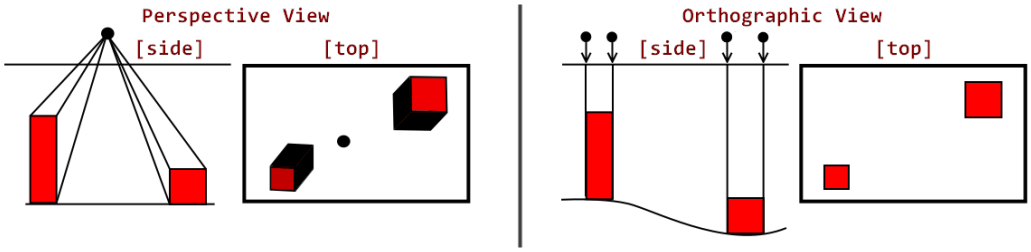

Figure 16: Perspective and orthographic view above a scene.

visible, but the back and the feet disappear, because they are "vertically below" the head and the upper chest.

The orthographic projection eliminates most of the local minima that not real heads, but preserves the single important minimum at the head.

When pixels are orthographically projected they are actually being shifted along the imaginary line that passes through the image center and the pixel. This is the reason for the white areas, whose pixels have been "shifted away" towards the center. Here, it is important to notice that two or more pixels may get projected to the same orthographic position (a collision). In that case the value of the pixel with the smallest range is taken, because this corresponds to the scene object that is "vertically top-most".

5.4. Handling Non-Convex Structures

The "mud stains" noise could be effectively removed by applying the median filter. But occasionally the range images contains natural, coarse-grained structures which are valid local minima, but at different positions than the human head. Typically, such areas are the shoulders and the backpacks which both can stand out like a local minimum. These structures look like:

- steep peeks (valid local minima with small area)
- deep valleys or holes (areas which provide potential for peeks)
- large areas of missing data that emerge from the orthographic projection

This kind of peaks, holes and missing data make the human silhouettes in the range image appear non-convex and that is breaking our precondition of a one-to-one relationship between local minima and human heads.

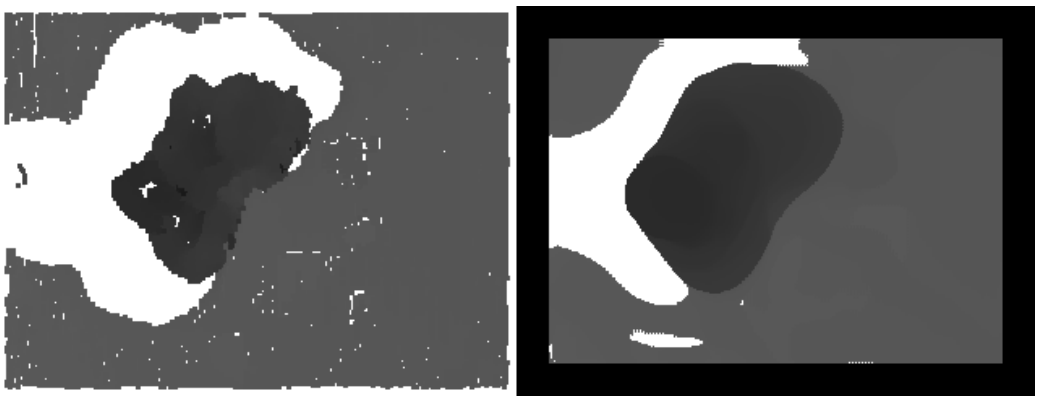

Figure 17: An orthographic range image (left) and its corresponding convexized image (right). Notice that the right image is also free of noise.

To solve this, we searched for a pattern for the human heads that was stable enough throughout a sequence of range images. But even in two subsequent images the heads looked slightly different because of random variations of the size and form of the head area and of the ranges at the head pixels.

So instead of fitting of the local minima to heads and doing a yes/no classification, we decided to *convexize* the range data before the head detection, so that only the real human heads appear as local minima after the convexization (Figure 15). After this step we could run the Pylon Grid detector and assume that all local minima will be human heads.

It turned out that we could convexize the landscape by simply applying our fast median filter, but with a *big radius* and taking the value at the 0.25 or 0.75 quantile instead at the 0.5 position. With this parametrization, the median filter has a *structure-altering side effect* which produces convex structures. Both 0.25 and 0.75 worked well, but the big radius of the filter (15-25 pixels) produced a strong morphological effect. Another interesting observation is that the convexization also removes noise, so no additional noise filtering is necessary.

While the fast median filter is still linear, now it has a big factor that slows it down. Speed can be gained here by sacrificing accuracy. This is achieved by looking only at some of the neighbor pixels in the kernel area of the filter (Figure 19.

In order to work quickly at this big radius, the filter sacrifices some accuracy by checking only every 4th pixel in the kernel.

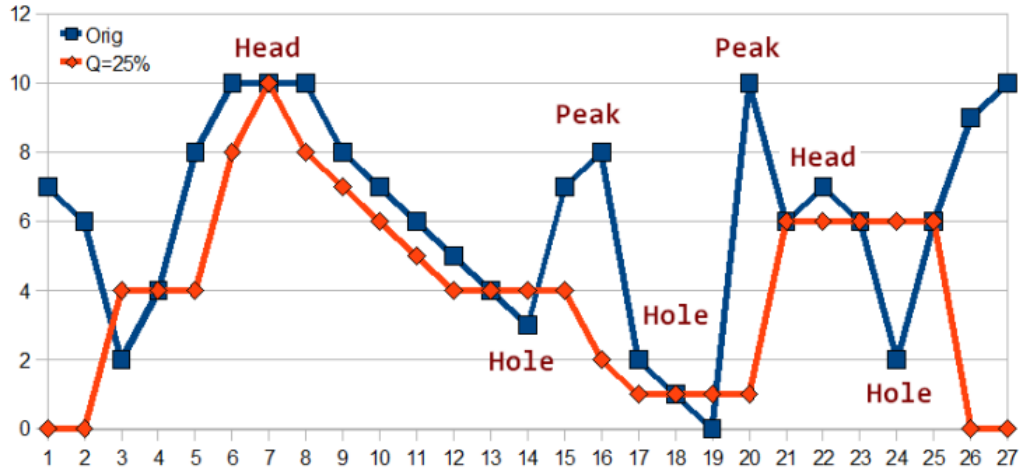

Figure 18: The median filter applied with quantille=0.25 and radius=2. This lowers significantly the peaks and rises a little the holes in the range image, effectively removing some sharp (false heads).

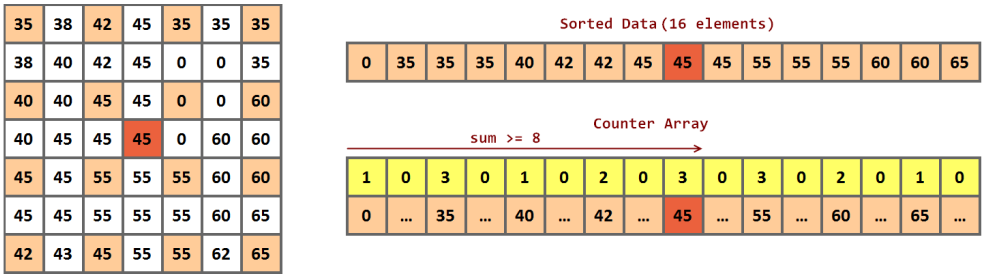

Figure 19: The kernel of the filter can be large, but sparse for gaining speed in a big radius.

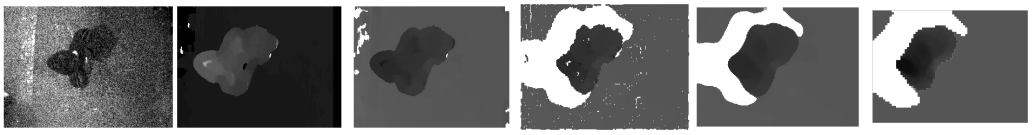

Figure 20: The stereo image goes through numerous transformations until it reaches its final form. $RAW Stereo \mapsto Disparity \mapsto Range \mapsto Ortographic \mapsto Convexized$.

After the convexized image is calculated the range data is no longer transformed. Now it has a shape that is very good for head detection and the preprocessed image is given as input to the Pylon Grid.

6. Results and Discussion

The Pylon Grid detector was as a core component in a the people counting system that we build using the Bumblebee2 stereo camera and the C# programming language. The performance of the system was tested in three outdoor scenes:

- camera height 5.2 meters, daylight, with sub-pixel interpolation turned on
- camera height 3.4 meters, daylight
- camera height 3.4 meters, at night with some artificial light

The camera was manually calibrated to the scenes and the range images we extracted with a proprietary library from by Point Grey Research.

6.1. Experimental Results

We present some snapshots of the system in action where people are tracked and counted. The pictures contain *hard situations* that are handled by the system. The peoples paths are shown and their heads are marked with bigger points at the end of the paths. The comments below the images state the ratio: *detected heads / fully visible heads*.

The mismatch of human heads to local minima occured in about 8 percent of the experimental images, mostly in hard situations with close formations of people.

6.2. Hard Situations

A situation is hard for human head detection when it is hard to segment the people from the background (occlusions among the range silhouettes) or to isolate individual people from each other (multiple range silhouettes merge one bigger silhouette).

The proposed scheme effectively eliminates the problem of oclusion by using a vertical camera orientation. Dense formations of people are not harder than sparse formations as long as the people don't have physical

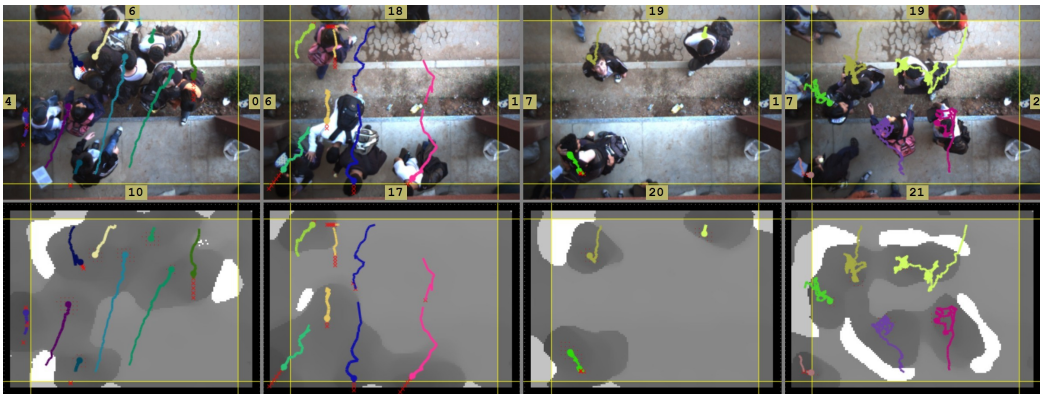

Figure 21: (1) - high density, 7/8; (2) - dense scene, 5/5; (3) - sparse scene, 3/3; (4) - dense scene, 6/8;

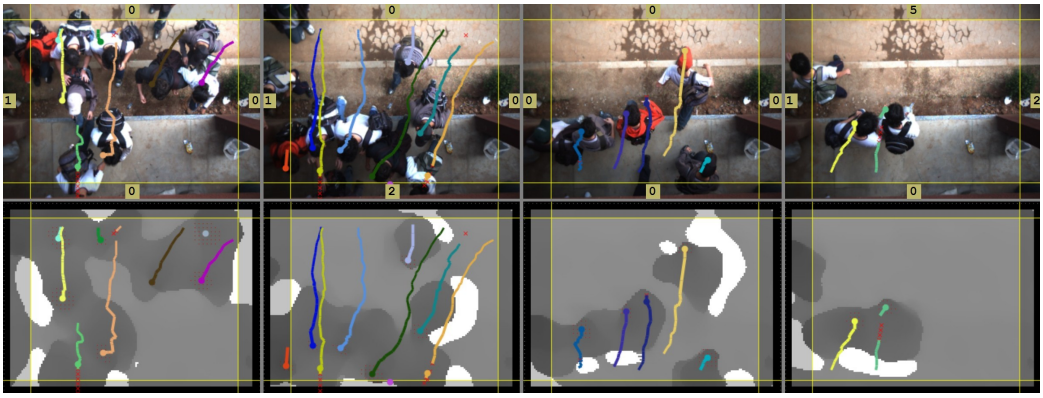

Figure 22: (1) - highly density, 7/8, one false head (the backpack); (2) highly dense scene, 6/6; (3) - dense scene, 4/5, one false head (the hand), (4) - normal scene, 2/3, person at the border is not detected.

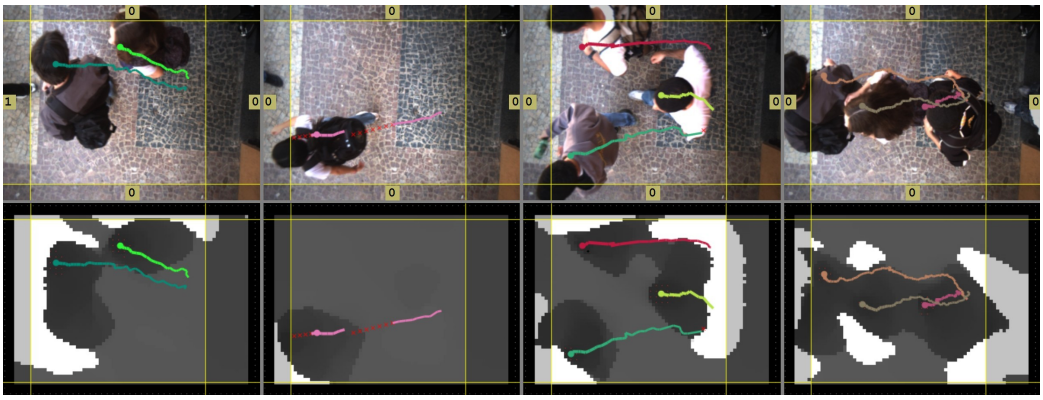

Figure 23: (1) normal density - 2/2; (2) - low density, 1/1; (3) - normal density, 3/3; (4) - high density, 3/3.

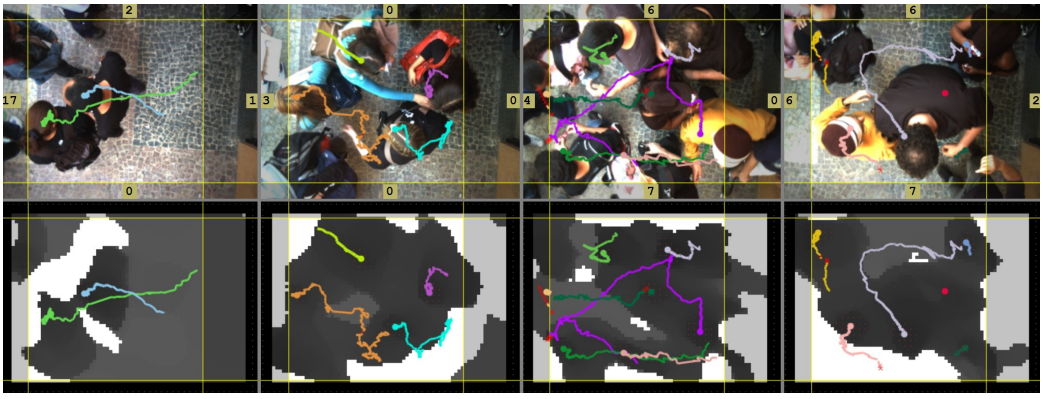

Figure 24: (1) - high density, 2/2; (2) - high density, 4/4; (3) - high density, 7/7; (4) - 4/4. one false head at the shoulder, one false head at the rised hand.

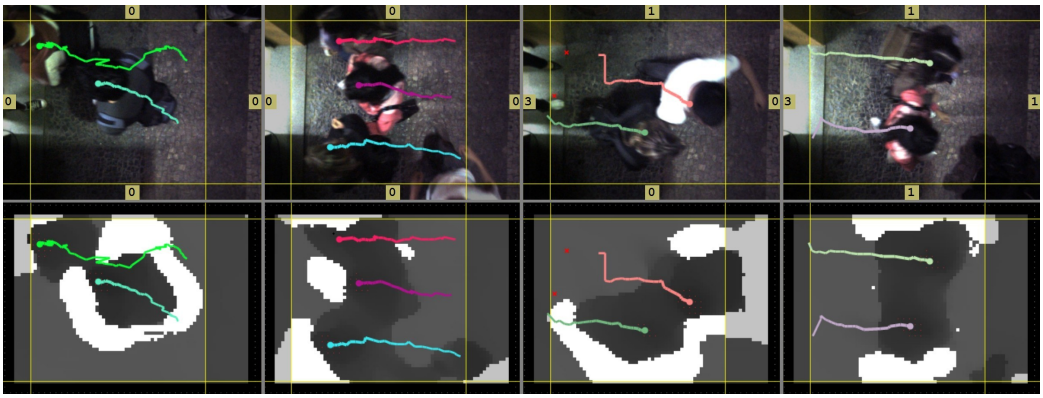

Figure 25: (1) - normal density, 2/2; (2) - normal density, 3/3; (3) - normal density, 2/2; (4) - normal density, 2/2.

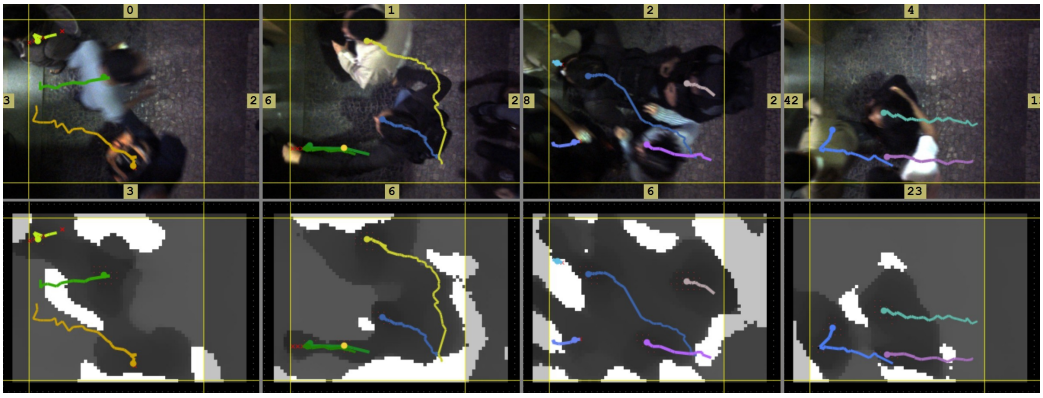

Figure 26: (1) - high density, 3/3; (2) - normal density, 2/2, one false head at the arm; (3)- high density, 4/4; (4) - high density, 3/3.

contact. If two people have physical contact their range silhouettes merge to a bigger silhouette. Now a problem emerges if the two people differ greatly in height, so that the merged silhouette shows only one local minimum after the preprocessing and the detector would detect the head of the taller person. This happens because the convexization of the range data might eliminate the local minimum of the smaller person. In the case where the two people don't vary too much in height the merged range silhouette correctly presents two local minima after the convexization.

Only in dense and chaotic scenes where multiple people had physical contact and moved in random directions, two local minima fusioned to a single local minimum and the head got lost. False heads were detected in some cases, when a person rose his hand or had a big backpack. The method performed very well in the night scene, where lightning varied but the quality of the range data was still good.

6.3. Robustness and Speed

With the orthographic projection, followed by convexization and noise filtering of the range data, the humans can be distinguished by one very stable feature the local minimum, which becomes unique for every head. From this point, the pylon grid algorithm can be used to find all local minima accurately and quickly.

The method is well suited for real-time applications because it is linear with respect to the pylons and has a predictable maximal runtime. In our complete counting system, where the Pylon Grid detector was just one component, we could continuously process 320x240 images at 15 FPS on a 2.4 GHz AMD Athlon processor and no use of multi-threading. Isolated tests of the Pylon Grid alone reached 90 FPS at a 640x480 resolution.

The method is also well suited for parallel processing since the detector is stateless and operates only in the context of a single frame. A simple and safe strategy would be to paralelize at the frame level - buffer as many frames as CPU cores available and supply each frame to a separate detector object.

6.4. Fast moving People

In a situation with fast moving people, there will be two additional effects:

- positions of the people will be shifted in subsequent frames
- range data might be less accurate and might contain more noise

The first point will challenge the *tracking* of the people, because the longer shift will increase the probability of mismatching a person in two subsequent frames. Tracking and matching is out this detector's scope. It is designed to locate the human heads in a *single* range image and keeps no internal state, does not learn and uses no information from any previous frames. The tracking problem can be approached with a more sophisticated component which uses the detector for each frame and then decides how to match and track the people.

The second point will challenge the *preprocessing* of the range data, because the detector will detect all local minima that are covered by a pylon, regardless of the quality of the input data. Since the accuracy of the detector depends on the one-to-one relationship between local minima and human heads, it is essential capture adequate range data and effectively preprocess it, so that noise is removed and human shapes are convexized. Fast moving people require video capturing with a fast frame rate, in order to have a sharp image with little noise. This problem can be approached with a good combination of scene illumination, optical system, and stereo extraction algorithm.bIn the case where noisy range data is supplied, we apply the preprocessing techniques described above. One of the experiments included a scene where a student was running with speed of about 5 meters per second and was captured at daylight from 5.2 meters above the ground at 15 FPS with the Bumblebee2 stereo camera. The student was detected correctly at each of the three frames where he appeared. The detector performed well, mainly due to the capability of the Bumblebee2 camera to extract good range data at these conditions.

7. Conclusion

This work confirms that range images captured from a vertically oriented camera provide an excellent base for pedestrian detection and tracking. It also shows that head detection can be performed efficiently and accurately, solely by the analysis of 3-D range data in a context-independent manner by examining only the individual frame.

The work contributes a fast and accurate Pylon Grid algorithm with linear complexity for local minima detection in range images. The algorithm is guaranteed to find all local minima and its speed only depends on the resolution of the grid. We also contribute a fast and flexible median filter with linear complexity for noise filtering and data convexization. The filter

can operate with a big radius, pick a custom quantile and trade accuracy for speed.

8. Acknowledgements

We want to thank Prof. Guilherme A. S. Pereira from the Department of Electrical Engeneering at the Universidade Federal de Minas Gerais for lending us the Bumblebee2 stereo camera from his laboratory. Another thanks to the excellent support team of Point Grey Research, who helped us resolve very specific problems regarding offline processing of the stereo footage.

Special thanks to Anderson Will Fortunato for helping us with the experiments. Special thanks to Gabriel Machado Fonceca and his lovely family for giving Rusi a home during his stay in Belo Horizonte.

[1] D. Conte, P. Foggia, G. Percannella, F. Tufano, M. Vento, A method for counting moving people in video surveillance videos, EURASIP J. Adv. Signal Process 2010 (2010) 5:1–5:10.

[2] S. M. Khan, M. Shah, Tracking multiple occluding people by localizing on multiple scene planes, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2009) 505–519.

[3] T. Zhao, R. Nevatia, B. Wu, Segmentation and tracking of multiple humans in crowded environments, IEEE Trans. Pattern Anal. Mach. Intell. 30 (2008) 1198–1211.

[4] D. Ramanan, D. A. Forsyth, A. Zisserman, Tracking people by learning their appearance, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2007) 65–81.

[5] D. B. Yang, H. H. González-Baños, L. J. Guibas, Counting people in crowds with a real-time network of simple image sensors, in: Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2, ICCV '03, IEEE Computer Society, Washington, DC, USA, 2003, pp. 122–.

[6] A. Albiol, M. J. Silla, A. Albiol, J. M. Mossi, Video analysis using corner motion statistics, in: Proceedings of the IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, 2009, pp. 31–38.

[7] O. Masoud, N. P. Papanikolopoulos, S. Member, A novel method for tracking and counting pedestrians in real-time using a single camera, IEEE Transactions on Vehicular Technology 50 (2001) 1267–1278.

[8] S.-Y. Cho, T. W. S. Chow, C.-T. Leung, A neural-based crowd estimation by hybrid global learning algorithm 29 (4) (1999) 535–541.

[9] M.-T. T. Hoang, Y. Won, A marker-free watershed approach for 2d-ge protein spot segmentation, in: Proceedings of the 2007 International Symposium on Information Technology Convergence, ISITC '07, IEEE Computer Society, Washington, DC, USA, 2007, pp. 161–165. doi:http://dx.doi.org/10.1109/ISITC.2007.8.
URL http://dx.doi.org/10.1109/ISITC.2007.8

[10] C. Rambabu, I. Chakrabarti, An efficient hillclimbing-based watershed algorithm and its prototype hardware architecture, J. Signal Process. Syst. 52 (2008) 281–295. doi:10.1007/s11265-007-0157-3.
URL http://dl.acm.org/citation.cfm?id=1388442.1388447

[11] S. Stiene, K. Lingemann, A. Nuchter, J. Hertzberg, Contour-based object detection in range images, in: Proceedings of the Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06), 3DPVT '06, IEEE Computer Society, Washington, DC, USA, 2006, pp. 168–175. doi:http://dx.doi.org/10.1109/3DPVT.2006.46.
URL http://dx.doi.org/10.1109/3DPVT.2006.46

[12] J. Marquez, W. Ramirez, L. Boyer, P. Delmas, Robust ellipsoidal model fitting of human heads, in: Proceedings of the 2nd international conference on Robot vision, RobVis'08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 381–390.
URL http://dl.acm.org/citation.cfm?id=1787703.1787739

[13] D. Beymer, Person counting using stereo, in: Proceedings of the Workshop on Human Motion (HUMO'00), HUMO '00, IEEE Computer Society, Washington, DC, USA, 2000, pp. 127–.