A Keypoint Detector based on Visual and Depth Features

Levi O. Vasconcelos, Mario F. M. Montenegro, Erickson R. Nascimento
Department of Computer Science
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte, Brazil
Email: {leviovasconcelos,mario,erickson}@dcc.ufmg.br

***Abstract*—One of the first steps in numerous computer vision tasks is the extraction of keypoints. Despite the large number of works proposing image keypoint detectors, only a few methodologies are able to efficiently use both visual and geometrical information. In this work we introduce KVD (Keypoints from Visual and Depth Data), a novel keypoint detector which is scale invariant and combines intensity and geometrical data using a decision tree. We present results from several experiments showing that our methodology produces the best performing detector when compared to state-of-the-art methods, with the highest repeatability scores for rotations, translations and scale changes, as well as robustness to corrupted visual or geometric data. Additionally, as processing time is concerned, KVD yields the best time performance among methods that also use depth and visual data.**

***Keywords*-keypoint detector; RGB-D image; decision tree; information fusion;**

I. Introduction

Selecting a set of interest points in images has been an omnipresent step in a large number of computer vision methodologies over the years. A careful choice of interest points in an image may significantly reduce the effect of noisy pixels and identify regions rich in information, which allows for an effective description of the regions containing these points.

Moreover, the ever growing volume of data of different kinds, such as high resolution images, RGB-D data (composed of visual and three dimensional data) and the massive image repositories available in the web, makes the creation of effective keypoint detectors crucial for a large number of computer vision techniques. Such detectors make it possible to reduce the data search space, thus making the processing of such data a manageable task.

We present a novel RGB-D keypoint detection technique that combines both intensity and 3D information and is capable of handling the lack of illumination as well as dealing with substantial noise on both intensity and depth data, while maintaining a good computational efficiency. The detector shows robustness to motion disturbances, such as rotation, scale, and translational motion. Figure 1 illustrates the benefits of using our detector.

The detection and selection of a set of points of interest, to which we will henceforth refer to as *keypoints*, consist in looking for unique points located in discriminative regions of the image that will account for good *repeatability*, which in turn may lead to less ambiguity. There is a vast body of literature on keypoint detectors, of which [1], [2], [3], [4], [5] are well known representatives.

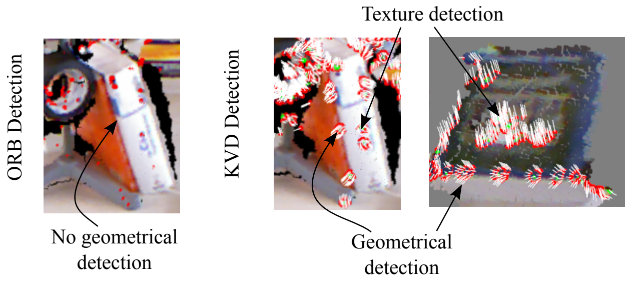


Fig. 1. Comparison between keypoints detected by our methodology (right images) and the ORB algorithm (left image). One may observe the absence of keypoints around the border of the book in the left image. Since KVD captures both visual and geometrical features to find keypoints it is able to detect such keypoints.

The richness of information being constantly engrafted in images has naturally pushed the envelope for several image based keypoint detection techniques. The vision literature presents numerous works that use different cues for keypoint detection based on pixel intensity. Keypoint detectors based on images alone seldom use other information such as the scene's geometry. As a consequence, common issues concerning real scenes, like variation in illumination and textureless objects, may dramatically decrease the performance of such techniques.

The newer 3D sensors are less sensitive to illumination than 2D cameras and provide the scale associated to each point. On the other hand, they are still very expensive and demand substantial engineering effort to be acquired. With the recent introduction of fast and inexpensive RGB-D sensors, the integration of synchronized intensity (color) and depth has become easier to obtain. The growing availability of inexpensive, real time depth sensors, has made depth images increasingly popular, inducing many new computer vision techniques.

Contributions: The main contribution of this work [1] is a novel keypoint detector called KVD (Keypoints from Visual

[1]This work is the result of a M.Sc. dissertation

and Depth data) that efficiently fuses intensity and depth data. By addressing keypoint properties such as distinctiveness, locality and efficiency, our methodology produces the best performing detector by using both visual and geometrical data, and presents a good performance and graceful degradation even in the absence of either one of them.

II. Related work

The detection of keypoints in images is an essential component in a myriad of applications in pattern recognition and computer vision algorithms. Since the seminal paper of Moravec [6], several keypoints detectors were proposed.

A recent approach that has become popular is based on machine learning techniques. Rosten et al.[4] proposed the Features from Accelerated Segment Test (FAST) detector, which extracts a simple descriptor based o intensity differences and makes use of a decision tree for classification. The work of Rublee et al.[5] presented the detector called Oriented FAST and Rotated BRIEF (ORB) that builds upon FAST's methodology. Which also implements a scale pyramid using the Harris cornerness function to achieve scale invariance.

Extracting data from images can usually provide rich information on the object features, but geometrical information produced by 3D sensors based on structured lighting or time of flight is less sensitive to visible light conditions. Three-dimensional data has been successfully exploited by Steder et al. [7]. The authors proposed the NARF descriptor to extract features from 3D point cloud data for object recognition and pose estimation.

The work of Holzer et al. [8] makes use of a random forest to approximate a specially tailored response function, which takes advantage of the curvature and the repeatability of each point.

Most recently, the algorithms on combining geometrical and visual have emerged as promising and effective approaches to deal with the tasks of object detection [9], Super-Resolution [10] and keypoint description [12]. Although several detectors exists for different information types, very few studies have addressed resolving the geometrical and visual fusion on low level features. To the best of our knowledge, only the unpublished work HARRIS6D, which the implementation can be found at PCL library [13], combines different information types to perform keypoint detection. In this work we build a new keypoint detector algorithm where both visual and depth data are used and we show that it effectively merges both information data with a low computational cost and yields high repeatability rate.

III. Methodology

Similar to the works of Rosten et al. [4] and Rublee et al. [5], our method is also based on a machine learning approach for keypoint detection. However, differently from these works, our approach has been designed to use both geometrical and visual data for improving the detection process and working even in the absence of image data.

The input to our algorithm is a data pair $(\mathbf{I}, \mathbf{D})$, which denotes the output of a typical RGB-D device, where $\mathbf{I}$ and $\mathbf{D}$ are the intensity and the depth matrices, respectively. Let $\mathbf{x} = (i, j)$ denotes a pixel's coordinates, $I(\mathbf{x})$ the pixel's intensity, $D(\mathbf{x})$ is the depth for that pixel, $P(\mathbf{x})$ is the corresponding 3D point, and $N(\mathbf{x})$ is its normal vector.

As stated, our technique is built upon a supervised learning approach, with a training step where a decision tree is created. This decision tree plays a key role in the detection procedure, since it is used to classify points into keypoints. There are three steps in this classification process: the feature vector composition, the model training, and the non-maximal suppression.

A. Feature Vector Composition

The first step of the detection process is to create a feature vector for every point, which will be fed into a decision tree for classification. Figure 2 shows a representation of the feature vector construction.

To decide if a given pixel is a keypoint, we analyze a circular vicinity of the given point in increasing radii. Figure 2 shows an example for the radii set $\mathcal{S} = \{3, 5, 7, 9\}$, which is the settings used in our experiments. It is worth to note that the algorithm is defined for any arbitrary set $\mathcal{S}$.

Given an image pixel coordinates $\mathbf{c} \in \mathbb{R}^2$, we consider its vicinity as the image patches that contain the circles centred at $\mathbf{c}$ with radii varying in $r \in \mathcal{S}$. Each circle is defined by the function $B(r, \mathbf{c})$ which we denote as $B_r(\mathbf{c})$:

$$B_r(\mathbf{c}) : \mathbb{R}^3 \rightarrow \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n\}. \tag{1}$$

The $B_r(\mathbf{c})$ map function outputs all pixels $\mathbf{p_i}$ lying within the Bresenham's circle [14] with radius equals to r. Thus, the total vicinity considered consists of the concatenation of all vectors $B_r(\mathbf{c}), \forall r \in \mathcal{S}$. We define the vicinity of a central pixel $\mathbf{c}$ as

$$\mathcal{V}_c = \{B_{r_1}(\mathbf{c}), B_{r_2}(\mathbf{c}), \dots, B_{r_{|\mathcal{S}|}}(\mathbf{c})\}, \quad \forall r_i \in \mathcal{S}. \tag{2}$$

For each vicinity element $\mathbf{p} \in \mathcal{V}_c$, we compute visual and geometric features.

Feature Extraction: The visual features are computed based on simple intensity difference tests among the vicinity. For each pixel $\mathbf{p} \in \mathcal{V}_c$ we evaluate

$$\tau_v(\mathbf{c}, \mathbf{p}) = \begin{cases} 2, & \text{if } I(\mathbf{p}) - I(\mathbf{c}) < -t_v \\ 1, & \text{if } I(\mathbf{p}) - I(\mathbf{c}) \geq t_v \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where t_v represents a tolerance threshold ($t_v = 20$ in our settings). This function analyses the intensity relationship between both pixels $\mathbf{c}$ and $\mathbf{p}$, encoding whether the pixel intensity $I(\mathbf{p})$ is darker, lighter or at similar intensity regarding the pixel intensity $I(\mathbf{c})$, respectively.

The visual feature extraction implemented by the function τ_v is similar to the one used by Rosten et al [4], however we embed geometric cues into our feature vector to increase robustness to illumination changes and to the lack of texture in the scenes.

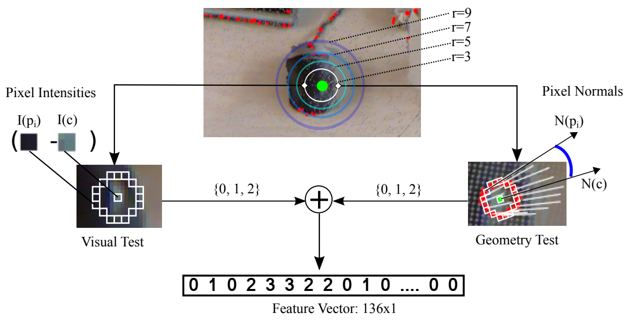


Fig. 2. Visual and geometrical feature extraction for a keypoint. The highlighted squares correspond the Bresenham's circle.

Geometric feature extraction is performed by the $\tau_g(\cdot)$ function. This process is depicted in Figure 2. It is based on two invariant geometric measurements: the normal displacement and the surface's convexity.

While the normal displacement test is performed to check whether the dot product between the normals $N(\mathbf{c})$ and $N(\mathbf{x_i})$ is smaller than a given displacement threshold t_g, the convexity test is accomplished by the local curvature indicator, κ, estimated as:

$$\kappa(\mathbf{c}, \mathbf{p}_i) = \langle P(\mathbf{c}) - P(\mathbf{p}_i), N(\mathbf{c}) - N(\mathbf{p}_i) \rangle, \tag{4}$$

where $\langle \cdot \rangle$ is the dot product, and $P(\mathbf{c})$ is the $3D$ spatial point associated with pixel $\mathbf{p}$ and depth $D(\mathbf{p})$. The κ function is used to capture the convexity of geometric features and also to unambiguously characterize the dot product between surface normals. The sign of the function κ denotes whether the surface between the two points assessed is convex (positive signed) or concave (negative signed).

Thus, the geometrical features are computed by analyzing the behavior of the surface between two points:

$$\tau_g(\mathbf{c}, \mathbf{p}_i) = \begin{cases} 2, & \text{if } \langle N(\mathbf{p}_i), N(\mathbf{c}) \rangle < t_g \wedge \kappa(\mathbf{c}, \mathbf{p}_i) > 0 \\ 1, & \text{if } \langle N(\mathbf{p}_i), N(\mathbf{c}) \rangle < t_g \wedge \kappa(\mathbf{c}, \mathbf{p}_i) < 0 \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Intuitively, it encodes whether the surface between the central pixel $\mathbf{c}$ and the queried point $\mathbf{p_i}$ is convex, concave or plane, respectively. We consider a plane if $\langle N(\mathbf{p}_i), N(\mathbf{c}) \rangle \geq t_g$. With $t_g = 0.97$ in our settings.

Scale Invariance: Scale invariance is endowed to our detector by taking advantage of the geometry information available, in order to weight the Bresenham's circles of different radii. We analyze the geometrical vicinity encompassed by each Bresenham's circle $B_r(\mathbf{c})$ in the 3D scene, by computing the minimum Euclidean distance among all vectors $\mathbf{v} = P(\mathbf{c}) - P(\mathbf{p_i})$, with $\mathbf{p_i} \in B_r(\mathbf{c})$:

$$d_r = \min_{\mathbf{p_i}} \|P(\mathbf{c}) - P(\mathbf{p_i})\|, \tag{6}$$

where $P(\mathbf{c})$ and $P(\mathbf{p_i})$ are the 3D points corresponding to the central pixel $\mathbf{c}$ and the pixels composing the Bresenham's circle $\mathbf{p}_i \in B_r(\mathbf{c})$, and $\|\cdot\|$ stands for the $L2$-norm.

The minimal distance d_r is taken as an estimative of the circle's radius in the 3D scene. It is then weighted by the unidimensional-Gaussian function, in order to penalize circles which its estimated radii in the 3D scene are distant from $\mu = 0.02$ meters, a predefined radius which seems as a reasonable value for our application.

$$\mathbf{w_r} = \exp\left(-\frac{(\mathbf{u} - \mathbf{d_r})^2}{2\sigma^2}\right), \tag{7}$$

the standard deviation $\sigma = 0.011$ was empirically chosen, targeting a sufficiently narrow shape around the mean value.

The weighting procedure avoids the addition of noise by circles covering non-interesting areas, e.g. for points too far from the camera, larger circles might have to be strongly penalized.

Final Feature Vector: In order to combine the geometric and visual information in one final feature vector, we deployed two different approaches: A.

1) *The additive approach:* This method combines both cues by adding both visual and geometric feature vectors. We extract a feature vector from a Bresenham's circle of radius r centered at $\mathbf{c}$ as a row vector $\vec{\mathbf{v}}_r = \begin{bmatrix} f_1 & f_2 & \dots & f_{|B_r(c)|} \end{bmatrix}^T$ where each $f_i \in \vec{\mathbf{v}}_r$ is given by:

$$f_i(\mathbf{c}, r) = w_r * (\tau_v(\mathbf{c}, \mathbf{p_{i,r}}) + \tau_g(\mathbf{c}, \mathbf{p_{i,r}})), \tag{8}$$

where $\mathbf{p_{i,r}}$ is the ith element of the respective Bresenham's circle $B_r(\mathbf{c})$. The final feature vector $\vec{\mathbf{F}}$ is generate by concatenating all the feature vectors $\mathbf{v_r}$ as:

$$\vec{\mathbf{F}} = \begin{bmatrix} \vec{\mathbf{v}}_{r_1} & \vec{\mathbf{v}}_{r_2} & \cdots & \vec{\mathbf{v}}_{r_{|\mathcal{S}|}} \end{bmatrix}^T \quad \forall r \in \mathcal{S}. \tag{9}$$

2) *The concatenation approach:* This approach concatenates both feature vectors, where one encodes visual features and the other geometrical features. We will depict the visual vector $\vec{\mathbf{F}}_v$ construction, while the geometric vector $\vec{\mathbf{F}}_g$ follows a similar process. For each Bresenham's circle, a row feature vector $\vec{\mathbf{v}}_r = \begin{bmatrix} f_1 & f_2 & \dots & f_{|B_r(c)|} \end{bmatrix}$, is computed where each $f_i \in \vec{\mathbf{v}}_r$ is calculated as:

$$f_i(\mathbf{c}, r) = w_r * \tau_v(\mathbf{c}, \mathbf{p_{i,r}}), \tag{10}$$

then, the visual feature vector $\vec{\mathcal{F}}_v$ is defined as:

$$\vec{\mathbf{F}}_v = \begin{bmatrix} \vec{\mathbf{v}}_{r_1} & \vec{\mathbf{v}}_{r_2} & \cdots & \vec{\mathbf{v}}_{r_{|\mathcal{S}|}} \end{bmatrix}^T \quad \forall r \in \mathcal{S}. \tag{11}$$

The geometric feature vector $\vec{\mathbf{F}}_g$ follows the same logic, but applying the function $\tau_g(\cdot)$ instead of $\tau_v(\cdot)$ in Equation 10. Finally, the final feature vector $\vec{\mathbf{F}}$ consists of the concatenation of both visual and geometric vectors:

$$\vec{\mathbf{F}} = \left[\vec{\mathbf{F}}_v, \vec{\mathbf{F}}_g\right]^T. \tag{12}$$

Thanks to the fusion process, our method is able to detect keypoints using both visual and geometrical data. Figure 1 depicts detected keypoints exploiting the fused information.

B. Decision Tree Training

The decision tree is a fundamental part of the algorithm, it is used to decide whether a target point should be considered as a keypoint candidate or not. In order to build the tree, we select a sample of training points, which contains both keypoints and non-keypoints examples properly labeled.

We created a training set by extracting a total of $160,144$ (66% of total) points - the remaining points were used for the test set - from the RGB-D Berkeley 3-D Object Dataset [15], which is publicly available[2]. Both sets were equally divided into positive and negative samples. In order to define the threshold of the curvature for a positive keypoint, we computed the curvature of positive keypoints which were manually selected. We have found the value of 0.09 based on the average of these curvatures. Thus, all the points with curvature larger than 0.09 were labeled as positive samples for the keypoint class. To take visual features into account, we also add keypoints detected by ORB as positive examples.

C. Non-Maximal Suppression

For non-maximal suppresion, we rank the keypoint candidates (classified as positive by the decision tree) lying within a small image patch. The best ranked candidate is selected as a keypoint. For the ranking function, let

$$\mathcal{X}_{rc_k} = \{\mathbf{p_i} : \ \mathbf{p_i} \in B_r(c) \ \wedge \ (\tau_v(\mathbf{c}, \mathbf{p_i}) = k \ \vee \ \tau_g(\mathbf{c}, \mathbf{p_i}) = k)\}, \tag{13}$$

be the set containing all the pixels within a Bresenham's circle $B_r(\mathbf{c})$ whose geometry or visual feature has value k.

For each radius $r \in \mathcal{S}$, we calculate the partial response $\mathrm{R_p}(\mathbf{c}, r)$ as:

$$\mathrm{R_p}(\mathbf{c}, r) = \max_{\mathcal{X} \in \{\mathcal{X}_{rc_1}, \mathcal{X}_{rc_2}\}} \left(\frac{1}{|\mathcal{X}|} \sum_{\mathbf{x_i} \in \mathcal{X}} D_v(\mathbf{c}, \mathbf{x_i}) + \lambda D_g(\mathbf{c}, \mathbf{x_i}) \right), \tag{14}$$

where $D_v(\mathbf{c}, \mathbf{x}) = \mathrm{abs}(I(\mathbf{x}) - I(\mathbf{c}))$ and $D_g(\mathbf{c}, \mathbf{x}) = 1 - \langle N(\mathbf{x}), N(\mathbf{c}) \rangle$ computes the visual and geometrical responses respectively and λ is a factor used to define the contribution of the geometrical information into the response.

The final response is then defined as the maximum response among all radii:

$$\mathrm{R_f}(\mathbf{c}) = \max_r \big(\mathrm{R_p}(\mathbf{c}, r)\big), \forall r \in \mathcal{S}. \tag{15}$$

Equation 15 uses both the absolute difference between intensities and the normal surface angles for the pixels in the contiguous arc of the Bresenham's circle as well as the keypoint candidate to rank the maximal points.

Finally, we divide the image into smaller patches with size $w \times w$ (in this work we use $w = 5$). For each patch we select the pixel with the largest final response.

[2]http://kinectdata.com

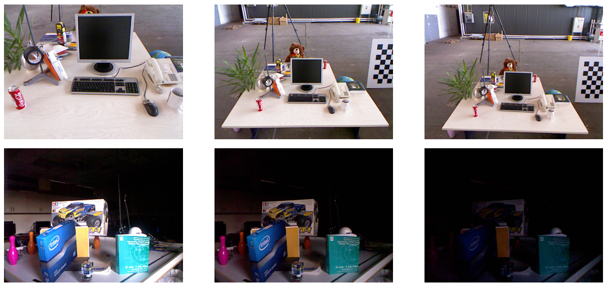


Fig. 3. Samples of the scale sequence (first row) and non-linear illumination transformations image sequences (second row).

IV. Experiments

We performed several experiments to evaluate the behavior of our detector. In order to analyze its repeatability, distinctiveness, robustness and time performance, we compared our approach against standard ones for two-dimensional images, SIFT, ORB, and SIFT3D (using all three color channels), for geometric data Harris3D (a 3D version of Harris corner detector) and NARF, and the Harris6D, which similarly to our methodology, uses both visual and geometrical data to detect keypoints.

We used the RGB-D SLAM Dataset presented by Sturm et al. [16] to evaluate the behavior of the methods regarding image changes in translation, scale, and rotation in both the image (roll) and horizontal (yaw) planes. This dataset is publicly available[3] and contains several real world sequences of RGB-D data captured with a Kinect$^{\mathrm{TM}}$sensor. Each sequence in the dataset provides the ground truth for the camera pose estimated by a motion capture system, which allows the computation of the homographies relating each image pair by a plane projective transformation.

In order to test the algorithms for illumination changes, we built a dataset by capturing a total of 104 images of a cluttered room starting at dusk (partial illumination) at an interval of one minute between acquisitions. The images were captured using a Kinect$^{\mathrm{TM}}$sensor with the resolution setted to 640×480 pixels standing at a fixed position, as shown in Figure 3.

A. Evaluation and Performance Metric

We evaluate and compare our method to others from literature regarding three concepts: Robustness, Distinctiveness and Time performance. In order to assess the Robustness of the methods, we use the Repeatability criteria.

Repeatability: The repeatability rate is computed as described by Mikolajczyk et al. [17], which computes the rate of matched keypoints among image pairs. A match is found if the intersection area $\mathcal{A}$ between two ellipses centred at the keypoint candidates is greater than a given threshold: $\mathcal{A} \geq \epsilon$. In this work we used $\epsilon = 0,4$.

Robustness: To test for robustness we evaluate the repeatability of the methods over several transformations: translational, scaling, rotational and illumination as well as artificial corruptions: contrast, brightness and gausian noise.

[3]https://cvpr.in.tum.de/data/datasets/rgbd-dataset

For each test, we compute the repeatability score under a specific image sequence $\mathcal{Q}_t = \{q_0, q_1, ..., q_n\}$, where each frame $q_i \in \mathcal{Q}_t$ along the sequence is increasingly affected by the targeted transformation. The repeatability score is calculated among all sequence pairs of the form (q_0, q_i), where $q_0, q_i \in Q_t$.

Distinctiveness and time performance: The keypoints distinctiveness evaluates the detector capability to find good features for a matching task for 2D descriptors, ORB and BRIEF [18], 3D descriptors, FPFH [19] and SHOT [18] and 2D+3D descriptors BASE [12] and CSHOT [11]. In order to evaluate the discriminative power of KVD detector, and to compare it against other approaches, we matched pairs of keypoints from several pairs of different images by using a brute force algorithm and feature vectors extracted by all these descriptors.

For time performance, we assess the processing time of the compared methods by averaging the detection time of 900 runs over entire images. The detection time was measured while the experiments were running on an Intel Core i7 3.5GHz (using a single core) processor running Ubuntu 12.04 64 bits.

B. Parameter Settings

In this section, we analyzed the best parameter values to be used by our detector. For this purpose, we used the RGB-D Berkeley 3-D Object Dataset [15].

We chose the radii set $\mathcal{S} = \{3, 5, 7, 9\}$ as it represents well the scale spectrum, given that the Kinect's reach for the depth image ranges from 0.3 to 5 meters. It showed a good balance between time performance and robustness.

In order to choose a value for the geometric threshold t_g, we ran the learning and testing phases for 15, 30 and 60 degrees. The value which returned the greatest accuracy in keypoint detection was the one with threshold for 15 degrees for the additive and concatenation combination.

We experimented with different methods for combining the geometric and visual features. Figure 4 shows the scale robustness evaluation for four different methods: additive (KVD), concatenation, texture only, and geometric only. We can see that fusing information yields a stronger keypoint detector. We chose the additive approach, since the concatenation spends twice the memory as the additive and both achieved similar accuracy (Figure 6). Also the additive approach shows better robustness to image corruptions noise (noise, brightness and contrast).

C. Results

We evaluated each detector for robustness to translational, scaling, rotational and illumination as well as artificial corruptions to contrast, brightness and added noise.

Figure 5 shows the results of the most meaningful tests. Our detector performs better than other approaches in most of the presented sequences.

An interesting result can be seen at the illumination change experiment Figure 5 (c). The figure shows that only KVD and HARRIS3D (which uses only 3D data), were capable to continue to provide keypoints under low illumination. Although HARRIS6D combines both geometric and visual cues like KVD, it reveals much more dependence on visual features than our proposed method.

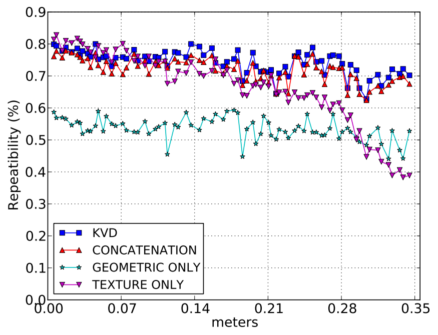


Fig. 4. Scale test evaluating our method for four different feature vector extractions. The image shows that combining both visual and geometric information yields a stronger keypoint detector.

As far as processing time is concerned, when comparing with other detectors which use geometrical data, KVD was the fastest approach. It processes 10^6 pixels per second, taking 0.06 seconds to process an image (and depth map) of size 640×480 pixels, while its main competitor, the HARRIS6D detector takes 0.08 seconds for images of the same size.

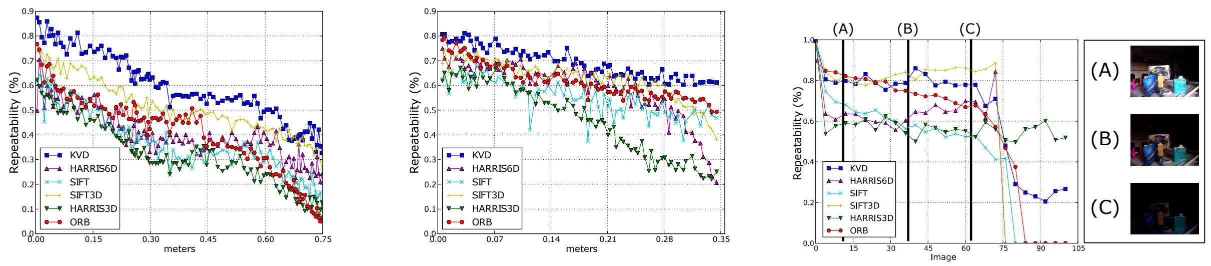
V. Statistical Analysis

In order to find the best algorithm and to evaluate the performance of our proposed method against others, we perform observation paired and the test of zero mean (using Confidence Intervals of 95%, as described in Jain [20]. The comparisons are shown at Figure 6. for CIs including the zero, it means that we can not infer which method performs better. Nonetheless, for CIs lying above or below the zero line means that KVD performed better or worse than the compared method, respectively. One can readily see that our method performed statistically better for the majority of the tests (exceptions for Contrast and Roll Rotation experiments, where KVD is outperformed by Harris 3D and SIFT 3D respectively). It is worth noticing that, despite the usage of more data by the concatenation approach (CAT label), it does not present better performance than the additive approach.

VI. Conclusion

In this work we proposed KVD, a keypoint detector capable of working with texture and geometrical data. A comparative analysis in terms of robustness to affine transformations, processing time and distinctness was conducted against the standard detectors in the literature.

Thanks to the strategy of combining different cues, our detector was more stable in the matching experiments. The combination of visual and geometry information indeed leads to a significantly better performance when compared to using either information alone. Moreover, our detector had similar processing performance and presented high repeatability scores for images severely corrupted with noise, images with



(a) Translational motion (b) Scale change (c) Non linear illumination

Fig. 5. Results for the repeatability experiment. (a) Horizontal translation motion; (b) Scale changing and (c) Nonlinear illumination change. Our method (KVD) is represented by the blue curve. We can see that, among other detectors using visual information, KVD can still provide reliable keypoints when visual image is deprecated.

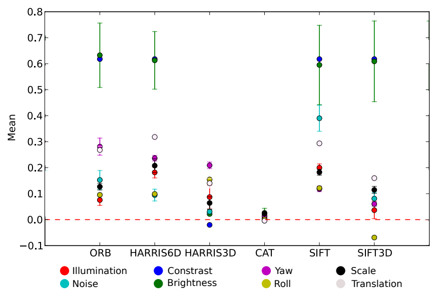


Fig. 6. We can see that KVD outperforms the others in the majority of the realized tests.

low contrast, saturated images, and several motion transformations.

Publications and Awards

- **Best Paper Award** Vasconcelos, Levi O., Erickson R. Nascimento, and Mario FM Campos. "A Scale Invariant Keypoint Detector Based on Visual and Geometrical Cues." Iberoamerican Congress on Pattern Recognition. Springer International Publishing, 2015.
- **Under review**: Special Issue on Pattern Recognition Letters (PRL).

Acknowledgment

The authors would like to thank Brazilian institutions CNPq, CAPES and FAPEMIG for partially financing this research.

References

[1] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the Alvey Vision Conference, AVC*, 1988, pp. 1–6.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. J. V. Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[4] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 105–119, 2010.

[5] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, 2011, pp. 2564–2571.

[6] H. P. Moravec, "Towards automatic visual obstacle avoidance," in *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, 1977, p. 584.

[7] B. Steder, R. B. Rusu, K. Konolige, and W. Burgard, "Point feature extraction on 3D range scans taking into account object boundaries," in *IEEE International Conference on Robotics and Automation, ICRA 2011, Shanghai, China, 9-13 May 2011*, 2011, pp. 2601–2608.

[8] S. Holzer, J. Shotton, and P. Kohli, "Learning to efficiently detect repeatable interest points in depth data," in *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Proceedings, Part I*, 2012, pp. 200–213.

[9] K. Lai, L. Bo, X. Ren, and D. Fox, "Detection-based object labeling in 3d scenes," in *IEEE Intl. Conf. on Robotics and Automation*, May 2012, pp. 1330–1337.

[10] D. B. Mesquita, E. R. Nascimento, and M. F. M. Campos, "Simultaneously estimation of super-resolution images and depth maps from low resolution sensors," in *Graphics, Patterns and Images (SIBGRAPI), 2015 28th SIBGRAPI Conference on*, Aug 2015, pp. 188–195.

[11] F. Tombari, S. Salti, and L. di Stefano, "A combined texture-shape descriptor for enhanced 3D feature matching," in *18th IEEE International Conference on Image Processing, ICIP 2011, Brussels, Belgium, September 11-14, 2011*, 2011, pp. 809–812.

[12] E. R. Nascimento, G. L. Oliveira, A. W. Vieira, and M. F. Campos, "On the development of a robust, fast and lightweight keypoint descriptor," *Neurocomputing*, vol. 120, pp. 141 – 155, 2013, Image Feature Detection and Description.

[13] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *IEEE International Conference on Robotics and Automation, ICRA*, 2011.

[14] M. L. Pitteway, "Algorithm for drawing ellipses or hyperbolae with a digital plotter," *The Computer Journal*, vol. 10, no. 3, pp. 282–289, 1967.

[15] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell, "A category-level 3-D object dataset: Putting the kinect to work," in *IEEE International Conference on Computer Vision Workshops, ICCV 2011 Workshops, Barcelona, Spain, November 6-13, 2011*, 2011, pp. 1168–1174.

[16] J. Sturm, S. Magnenat, N. Engelhard, F. Pomerleau, F. Colas, W. Burgard, D. Cremers, and R. Siegwart, "Towards a benchmark for RGB-D SLAM evaluation," in *Proc. of the RGB-D Workshop on Advanced Reasoning with Depth Cameras at Robotics: Science and Systems Conf.*, 2011.

[17] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. J. V. Gool, "A comparison of affine region detectors," *International Journal of Computer Vision*, vol. 65, no. 1-2, pp. 43–72, 2005.

[18] F. Tombari, S. Salti, and L. di Stefano, "Unique signatures of histograms for local surface description," in *Computer Vision - ECCV 2010, 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part III*, 2010, pp. 356–369.

[19] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *2009 IEEE International Conference on Robotics and Automation, ICRA 2009, Kobe, Japan, May 12-17, 2009*, 2009, pp. 3212–3217.

[20] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley- Interscience, 1991.