VIALAB machine vision benchmark: preliminary results

S. Giardino – F. Gori – C. Salati
Consorzio T3LAB, via Sario Bassanelli 9/11, 40137 Bologna
simone.giardino@t3lab.it – fabio.gori@t3lab.it – claudio.salati@t3lab.it

Abstract

The paper presents some preliminary results of the VIALAB benchmark for machine vision libraries, based on the five challenges that we have published in April 2012. The work that we have done allows also us to reconsider not only some of the technical decisions we had to make in the definition of the challenges that make up the benchmark, but also to reassess some of the principles on which it is based. We learned that a benchmark is a work that proceeds by successive approximations, and we feel that we are now ready to outline and discuss with all those that are interested in the topic the possible evolutions of the project.

Keywords


1 Introduction

The machine vision benchmark that is part of the VIALAB project [1] is different from academic benchmarks in that it is specifically aimed at commercial libraries and their overall characteristics: its main goal is to support users in the selection of the software that best fits their needs.

It was first presented to the machine vision community, and in particular to vision library vendors at Vision 2011 in Stuttgart [2][3] raising heterogeneous reactions.

The first set of challenges was made public in March 2012 in a dedicated page of the VIALAB project web site [5], raising again the interest of the machine vision community [4]. At the same time a document was released that states the ruling principles of the VIALAB machine vision benchmark [8].

Several results are now available and can be posted for discussion. But, more important, a first assessment can be made on the principles behind the VIALAB benchmark as well as on the challenges that we have defined.

Finally, we will discuss briefly the possible evolution of the benchmark project.

VIALAB
Tecnologie ed algoritmi per la visione artificiale
Bologna, Museo del Patrimonio Industriale – 23 gennaio 2013
2 A reassessment of the project

2.1 Collaborative strategy

The benchmark aims at comparing some of the most relevant, commercial machine vision libraries and, beside them, the open source, BSD licensed OpenCV [4] (not all of OpenCV is actually under BSD license for commercial purposes. This is specifically true for key modules of the detection and localization algorithms for textured objects: see also sec. 2.6). Based on a survey that VIALAB has performed among companies that use machine vision in the Emilia-Romagna Region [10][11], vendors, integrators, OEM, final users, we have identified from the very beginning a set of library vendors that are the primary target of our benchmark: this set includes Cognex, Matrox, MVTec and Teledyne/DALSA. However, from the beginning, we have also been open to consider any additional libraries that adhere to the rules of the benchmark [8].

BSD licensed OpenCV [4] will be benchmarked too, though it plays a special role in the project [8]. Document [8] states: “we will run the benchmark as a part of our project (VIALAB), but we think that the best way we can run it is in cooperation with library suppliers. The reasons for this are manifold: among others and most relevant is that each library supplier is likely the best user of its own toolkit, and we would like to compare the best possible solutions for each of the challenges of the benchmark.”

There is however another argument that lead us to look for the cooperation of library vendors: previous experiences have shown that the availability of well-defined challenge tasks with ground truth and evaluation metrics has succeeded well in advancing the state of the art in several research fields [12]: in this view our benchmark is supposed to benefit also library vendors.

Unfortunately this approach didn’t work. As previously stated, the reaction of library vendors at Vision 2011, even those that did not belong to the set of our primary targets, was heterogeneous, ranging from enthusiastic to absolutely uninterested or hostile. In practice, though, no vendor proved to be ready to invest enough resources in the project, so that we ourselves had to implement almost all solutions of the challenges for all libraries.

Even the comments that we received about the definition of the challenges focused mainly on technical details, however important, and sometimes only on details that could affect the results of the library of the person making the comment. So, even the overall definition of the benchmark’s challenges didn’t really benefit of vendors’ contribution: this lack of contribution has hampered our efforts in several aspects, as it will be clear in the following sections.

VIALAB asked also other institutes to contribute to the benchmark: unfortunately, even this didn’t work. An alternative approach is thus required (see sec. 4).

2.2 Metrics

In spite of our efforts, the definition of the metrics that we are considering is far from perfect [8].

First of all, resolution was not considered. Resolution was considered specifically relevant in a challenge whose definition hasn’t yet been completed, even though it was listed among the challenges that we wanted to define during the Vision 2011 presentation [3], and even though we have already created part of the dataset that will support it: object localization in real world coordinates (in 3D space); the part of the dataset that is missing is exactly that which is related to resolution.

Resolution would have been a relevant metric for three of the challenges that we have defined, but we didn’t pay enough attention to it.
ROC (Receiver Operating Characteristic) curves are the preferred evaluation tool used in academic research for object detection algorithms, therefore we decided to use them as well and, according to academic research, we chose the area under curve (AUC) as the overall performance index [9]. However, a common requirement of object detection tasks in the industry is to operate at very low FPR (false positive rate; usually, false positives are not acceptable at all): therefore the AUC value doesn’t probably represent the best metric in an industrial context, as the analysis should be focused on the leftmost part of the ROC space, where the FPR is low (as a matter of fact the performance data that we have collected emphasize this part of the ROC curves, e.g. through the values of “input score threshold” that we have used). On the other hand, the AUC value reflects a more general behavior of algorithms (e.g. its robustness), because it is computed over the whole range of FPR, so it still represents a significant metric even in the context of industrial solutions.

Because of its overall goals, the VIALAB benchmark takes into account other metrics beside those that are related to effectiveness and speed. One of these metrics is the ease of use, and we have related this metric to the software complexity of the code implementing a solution [8].

Our benchmark framework is based on C/C++, and we actually required all solutions of the challenges to be implemented in C/C++, therefore even software complexity is assessed based on C/C++ source code. However several libraries come with a development environment that makes the implementation of an application much faster, and actually avoids writing C/C++ code completely, to the point that, in the case of Cognex VisionPro, implementing a solution directly in C/C++ proved itself to be a challenge. Accordingly, the way we try to measure ease of use is probably not general enough.

Another metric related to the usability and efficiency of a library is interoperability, which we related to the possibility to easily exchange data (specifically, images) with OpenCV: in this context OpenCV plays the role of a proprietary library that an integrator might wish to use together with a commercial library. In practice the assessment of this metric is trivial: MVTec Halcon and Matrox MIL can interact with OpenCV at no cost for grayscale images, while we didn’t find any way to construct a VisionPro-like image starting from an OpenCV image: we don’t know what the situation would be if we used Cognex CVL instead of VisionPro, though we guess that the situation in this case would be similar to that of Halcon and MIL. Things might be different for color images but we restricted ourselves to grayscale.

The assessment of efficiency is largely related to the capability of the libraries to take advantage of the characteristics of the hardware platform. The hardware platform (quad core, hyperthreaded, x86 based PC with Windows7 and a 336 core GPU, a GeForce GTX 460, with 1GB DDR5) supports 3 processing configurations:

- single core,
- eight core (4 + 4, thanks to hyper-threading),
- eight core with GPU.

To our knowledge all considered libraries, MIL 9.0, VisionPro 7.2 and Halcon 10/11 intrinsically and transparently support parallel processing: parallelism can be disabled only with Halcon. Additionally MIL 9.0 and Halcon 10/11 support GPUs in some of their operators (a feature that in both cases must be explicitly enabled), but actually none of the operators that we are using.

This means that we can test Halcon both in the single core and the multicore configuration while MIL and VisionPro run only multicore. The effect of GPUs cannot be assessed in our challenges.

### 2.3 The challenges

The lack of cooperation with the library vendors has significantly affected the definition of the challenges and their significance.

Based on our familiarity with OpenCV and on our previous work, we thought that “2.5D” vision (recognition and localization of a 2D pattern in the 3D space) is a particularly relevant issue and that all libraries would support it and, consequently, would support 3D calibration. In fact the prototype challenge that we presented at Vision 2011 together with the outline of the benchmark was camera calibration in the 3D
space [3]. At that time we didn’t receive any specific feedback about this challenge, which was later part of our first set of official challenges, which included also 2D object detection in orthogonal view images and under perspective distortion, and 2D object localization in orthogonal view images and under perspective distortion [5].

It turned out, though, that, between the libraries that we have considered (see sec. 3), camera calibration in the 3D space is supported, in the way we defined the challenge, only by MVTec Halcon.

Cognex supports this functionality as part of a different product (VisionPro 3D) than the one we have considered (VisionPro; see also sec. 3).

Matrox MIL actually supports it, but based on procedures that are different from the one that we have mimicked from OpenCV. The first procedure supported by MIL (M_TSAI_BASED, based on the Tsai algorithm [6] relies on a single calibration image that must cover the whole FOV (field of view) of the camera and that must have a minimum inclination of 30° relative to the image plane; the calibration procedure envisioned in our challenge, and for which we collected images and coordinates of calibration points is based on multiple images, each of which may thus cover only a small part of the field of view [7]: trying to apply the MIL calibration procedure to a single image of our calibration set turns out unfeasible as it yields localization errors that are 200 times larger than those we measure with OpenCV and Halcon. The second procedure supported by MIL (M_3D_ROBOTICS), typical of robotic applications where a common reference system must be defined for the vision and robotics subsystems, requires that the world coordinates of all calibration points of all images are expressed in a same reference system [7], a constraint that is not present in the procedure that we have envisioned and for which we have collected measurement data.

As a consequence of all this, we could actually apply our challenge definition only to Halcon. A better cooperation of library vendors would have avoided these problems.

All considered commercial libraries support 2D camera calibration (Halcon as a special case of 3D calibration); the reason is that from the standpoint of nowadays industrial application 2D calibration is still more important than camera calibration in the 3D space: for example, items moving on a conveyor lie on a plane. Unfortunately, being focused on camera calibration in the 3D space we didn’t collect images and coordinate measurements capable to support a 2D camera calibration challenge.

We faced some difficulties even when we tried to apply our challenge definition to Halcon, even though MVTec had reviewed it. One of our evaluation procedures, as explained later (sec. 3.1), is based on forward projection (see section 2.4) and relies on the availability in the library of a procedure capable to identify and localize all intersections of a checkerboard without knowing its position and orientation (likewise OpenCV): unfortunately such a procedure is not available as such in Halcon, and doing an accurate and robust implementation of it would have required too much of an effort (a corner detector identical to that used by OpenCV is available in Halcon, but MVTec questions the accuracy one can reach based on it: again, this highlights a weakness in the definition of the challenge that could have been avoided with the help of vendors). For this reason we limited ourselves to the other evaluation procedures that had been envisioned in the challenge definition.

Our approach proved to be perhaps too academic also with respect to the other four challenges of the first set [5].

Considering the two challenges involving orthogonal view images, we placed excessive emphasis on the handling of scaling: the dataset is based on images, real and synthetic, where the size of the target object may vary, and object detection procedures are invoked so that they can recognize objects with a scaling factor in a range from 0.75 to 1.25.

After this was noticed, we set out to define a new localization challenge that would avoid scaling, a condition that is considered “too easy” in current state-of-the-art scientific work but turns out more common in industrial applications. Again, we could not base this new challenge on the dataset that we had created for the initial 2D object localization challenge in orthogonal view images, but because this dataset consisted completely of synthetic images creating a new dataset was feasible.
In industrial applications with non orthogonal view, perspective deformations would probably be irrelevant because 2D calibration would compensate for it. However, because we considered the case in which no calibration is performed and therefore no image rectification is possible, the two challenges of 2D object detection and localization under perspective distortion may make sense: we further generalized the case because, even this time, we allowed for scaling (which, again, is probably not very relevant in many current industrial applications). We acknowledge that in the case of the considered libraries these two challenges are much less relevant than the two based on orthogonal view.

2.4 Ground truth

Associating a reliable ground truth to thousands of images is a huge work, and a significant part of our effort was spent on it, implementing a framework that could automate the job. The problem of the definition of the ground truth, however, is not only related to the amount of work it may require.

Several types of ground truth must be considered. In object detection challenges the ground truth is relatively easy (though some localization check is required to verify that the object was really correctly detected). This is not the case when localization information is part of the challenge: in the case of our challenges this occurs in three cases, related to object localization in the image space and to the calibration challenge.

In the calibration challenge [13] three evaluation procedures have been envisioned, one based on back projection (we provide the real world coordinates of a relevant point, the solution returns its position in the image and we compare it with the ground truth information) and two on forward projection (we provide the image coordinates of a relevant point and its distance from the camera, the solution returns its position in the real world: the ground truth consists of this coordinates measured in a test-bench): see sec. 3.1 for additional details.

The accuracy of our measurements in the real world reference system leads to an accuracy of absolute coordinates that is about 150μm, which applies to the data we feed into the calibration procedure as well as to our ground truth information: thus, we cannot assess any library that performs like that or even better than that; apparently all libraries do that.

This accuracy looks coarse, but one must take into account that it is related to coordinates in an absolute reference system: errors may come from several sources, such as defects of the physical edges of the target and the accuracy of the mechanical and optical test benches involved in the measurements of these absolute coordinates [13]. Working in an arbitrary reference system or with differential measures would have increased accuracy, but we wanted to somehow mimic the complexity of a real application setup where the reference systems of all components (e.g. vision system and robotic arm) must be coordinated. We think that differential measures should be part of another challenge, not related to localization but to dimensioning of objects in the world space.

Our real world ground truth information is inadequate, but this actually highlights the difficulty that one faces when she/he wants to realize a highly accurate application: the limit is not given by the library but by the quality of the calibration target and by the measurement system of the real world.

Another difficulty is related to the definition of the ground truth information in the image space: how to determine the real position of a relevant point in the image? We ended up using a procedure based on OpenCV (relevant points are actually the intersections of a checkerboard: their detection is also the base of the calibration procedure of OpenCV): thus, we define the ground truth using the same tools that we want to evaluate using this ground truth (besides, the accuracy of the OpenCV corner detector, based on the gradients method, especially under perspective distortion, has been questioned). A variation of this, would have been using the average of the values returned by all libraries (this would have actually been impossible, see section 2.3). To improve somewhat the reliability of our ground truth the only thing we actually did was to compute the position in the image of a relevant point based on several images of the same scene.

Based on this considerations, we took a different approach in the challenges related to object localization in the image space [15][16]: in this case we decided to use synthetic images where the ground truth is known
a priori. We are well aware of the fact that by applying a homography to an image we are introducing interpolation errors and Table 1 shows the effect of applying a homography and its inverse on the gray value of the pixels of an image when different interpolation techniques are used. Of course, in our benchmark we used Lanczos interpolation. Based on our experience (see section 3) these errors don’t play any significant role in the benchmark procedure.

<table>
<thead>
<tr>
<th>Interpolation type</th>
<th>MSE (Mean Square Error)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nearest Neighbor</td>
<td>9,8475</td>
</tr>
<tr>
<td>Linear</td>
<td>6,8230</td>
</tr>
<tr>
<td>Cubic</td>
<td>3,8668</td>
</tr>
<tr>
<td>Area</td>
<td>6,8230</td>
</tr>
<tr>
<td>Lanczos</td>
<td>3,5457</td>
</tr>
</tbody>
</table>

Table 1: Effect of homographic transformation and interpolation on gray value

2.5 Synthetic images

The decision to make use also of synthetic images was taken from the very beginning [3] and we have repeatedly argued for it [8].

The idea of calibrated presence of nuisances would not have been feasible if it hadn’t been for the use of synthetic images: in fact we have spent a significant effort for the definition of nuisance models and transformations to apply to images and in their implementation in an image processing tool.

Using synthetic images has proved to be the right approach: not only it has allowed us to reach all our goals, but when we decided to introduce a new challenge (object localization without scale variations) to cope with the limits of the challenges that we had already defined, it allowed us to actually do it straightforwardly.

2.6 The object dataset

Several of our challenges are related to object detection: thus, one can ask what application field do we had in mind when we selected the objects that we used to build our image dataset.

As it is apparent from the way we describe our challenges, we are not addressing any specific application fields: objects are partitioned based solely on their property of being textured or not, as the types of algorithms that can be used to recognize these two categories of objects may vary significantly (e.g.
contour-based vs. descriptor-based): one can have an idea of the objects we are considering by looking at the development datasets of our detection and localization challenges [5].

Most of our untextured objects are mechanical parts, in general with a limited level of threedimensionality. Some of our textured objects can be related to electronic manufacturing (a PCB) or to the food industry. Figure 1 shows some object samples from the development dataset.

Trying to extrapolate from the overall results of the benchmark the suitability of one or the other library for a specific application, or a specific application domain is improper: these results are for instance normally averaged over a wide set of nuisance conditions that in a real world application one would try to eliminate. Additionally, as stated in [8], “not all the evaluation dataset will be available for development: this is because we want solutions to be general, i.e. capable of handling a class of problems, e.g. ‘detection of non-textured’ objects, and not only a list of specific cases.” This means that no application specific optimization is possible in our evaluation framework.

2.7 Solutions

As we have already stated, unlike our plans we had to implement most of the solutions for the challenge tasks. This obviously means that these solutions may be non optimal in spite of our best effort, and this, perhaps, also because of questionable design decisions.

An example of this is given by the solution we provided (uniformly, for all libraries) for the object detection challenge: although contrast polarity of the contour of all objects in the dataset is known, we decided to ignore this characteristic despite all libraries supporting the possibility to exploit it. Our rationale was to favor generality of solutions instead of effectiveness and efficiency: this approach is apparent also in the definition of this and other challenges: no polarity parameter was present in the template of the requested solution.

It has been pointed out that in the real world only a small minority of applications would take our approach: thus, it would have been better to provide and compare two solutions, one that takes contour polarity into account and one that does not. We have tried to deal with this point running some additional sample tests for which we present comparison data (see sec. 3.2).

2.8 Complementary results

According to a citation from [17], we stated in our policy statement that the results of the benchmark will “include the code fragment used to solve the benchmark task. Beside transparency, this will allow users to learn more about the use of a given system” [8].

Currently, only the framework software and the example solutions based on OpenCV have been published [5] (as part of the definition of the challenges), but all solutions that have been implemented for all libraries will be published with the final results by the end of the project.

If we consider all material that was necessary to prepare for the definition of the challenges and the solutions that we had to develop in a way that was conformant to challenges’ definition, we achieved as a side effect several additional results: we are now considering the best way to make all these results available to all stakeholders (see also section 4).

The first, obvious result is the creation of a huge dataset (2 TB) of images with associated ground truth: this dataset has already been used by the VIALAB participants for their own developments.

Another byproduct is the software, based on OpenCV, that we have used for the generation of synthetic images with the associated ground truth, supporting similarity and perspective transformations as well as introduction of a controlled amount of nuisances [8].

Also based on OpenCV we developed exemplary solutions and software for image acquisition: all this will be published as a VIALAB extension to OpenCV. It will include:

- A family of classes to handle 2D, grayscale cameras, based on an abstract class and containing also two instances for Allied Manta and IDS uEye cameras.
• An integrated calibration procedure based on multiple images of a single calibration target.

• A set of procedures, based on the SURF and Kd-tree primitives available in OpenCV, for the detection and localization of textured objects. Geometric validation is based on a 3 or 4 dimensional Hough pose space, depending on the fact that the view of objects is orthogonal or subject to perspective distortion. Pose estimation may be done coarsely, computing the coordinates of the object’s centroid, or accurately, either through an algorithm based on a least squares similarity estimation approach in case of orthogonal view images or through the use of an OpenCV primitive to estimate the homography in case of perspective view images.

• A set of procedures, based on the template matching primitive of OpenCV, for the detection and localization of untextured objects in orthogonal view images without scale variations.

• Procedures to compute, given the calibration parameters, forward (image → world) and backward (world → image) projection of N points (these procedures are actually library independent).

VIALAB will publish also utility procedures for other libraries:

• An integrated calibration procedure based on multiple images of a single calibration target for Halcon.

• Procedures for detection and localization of textured and untextured objects for Halcon, based on its shape based matching in case of orthogonal view images and on its deformable model matching in case of non orthogonal view images.

• Procedures for detection and localization of textured objects for Halcon, based on its descriptor-based matching, for both orthogonal and non orthogonal view images.

• A set of pairs (QuickBuild project, C++ stub class) that make VisionPro procedures for object detection and localization callable in the context of a Visual Studio 2010 solution (for both orthogonal and non orthogonal view images). This represents a significant example of how VisionPro procedures can be invoked in the context of a C/C++ program.

3 Preliminary results

The following sections present some preliminary results related to the first five challenges of the VIALAB benchmark.

In some cases we feel that our results need to be further discussed before we can disclose them even to a limited audience: this is the case for data related to efficiency but also for data related to the sub-challenges of detection and localization of textured objects.

Even the data that we present here must be handled with care: for example, if we consider object detection in orthogonal view images we see not only that a set of results can be interpreted in different ways (see Figure 11) but also that applying the solutions to different datasets yields different results.

The commercial libraries that we have considered are Cognex VisionPro 7.2, Matrox MIL 9.00R2 and MVTec Halcon 10/11. Whilst the choice for Matrox and MVTec was obvious, except perhaps for the product’s version (subject to evolution during the lifetime of our project), things were different in the case of Cognex, whose commercial offer is more articulated; the basic version of VisionPro that we considered is not targeted at 3D applications and does not include a procedure to support the solution of one of our challenges, 3D calibration of a single camera. Additionally, though providing a friendlier development environment, VisionPro restricts somewhat the capabilities of its computational engine, CVL (Cognex Vision Library), which can actually be acquired as a separate product: this created difficulties in another challenge. Our choice was based on the perception that VisionPro is the primary product. We are sorry for the limits of the choice we made, but again this is largely a result of the lack of cooperation of vendors.
3.1 **3D calibration challenge**

As we have seen, there are only two of the libraries that we considered that can provide a solution to this challenge, at least in the way we defined it: OpenCV and Halcon. Cognex deals with this problem in a different product and Matrox MIL’s 3D calibration procedures are different from that we have considered and for which we have collected data.

The best way to look at performance data about OpenCV and Halcon is to collect them as a function of the number of calibration images, averaging all other parameters. Analyzing data in this form shows general results that can be used when the work conditions are unknown, and is also a good method to highlight how the quality of calibration depends on the number of images used to perform it (see Table 2).

As indicated in [13] two methods have been used to evaluate the accuracy of the calibration procedure: backward projection and forward projection, and for this last method two procedures have been defined, one relying on the availability in the library of a function capable to identify and localize all intersections of a checkerboard, the other that instead deploys a function written by us using OpenCV to carry out this task. As already stated (see sec. 2.3), the first evaluation procedure based on forward projection could not be applied to Halcon. In a second time, also another back projection procedure has been introduced, similar to the first one but that computes the backward projection on the “image 0” (the calibration image used to find extrinsic parameters [13]).

<table>
<thead>
<tr>
<th>Number of calibration images</th>
<th>OpenCV Forward projection points (error in mm)</th>
<th>Halcon Forward projection points (error in mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.285</td>
<td>0.237</td>
</tr>
<tr>
<td>7</td>
<td>0.282</td>
<td>0.242</td>
</tr>
<tr>
<td>10</td>
<td>0.280</td>
<td>0.230</td>
</tr>
<tr>
<td>17</td>
<td>0.294</td>
<td>0.230</td>
</tr>
<tr>
<td>24</td>
<td>0.290</td>
<td>0.229</td>
</tr>
<tr>
<td>30</td>
<td>0.288</td>
<td>0.229</td>
</tr>
</tbody>
</table>

Table 2: Error according to different evaluation procedures
(as a function of the number of calibration images)

With respect to backward projection, where the error is measured in pixels, Halcon looks a little better than OpenCV: this figure of merit applies consistently, both when we consider results averaged over all camera resolutions and optics, and if we consider results that are specific to a camera resolution and optic. Still, these figures cannot be considered quantitatively significant, as highlighted by the following argument, and thus are not reported here.

When we consider the only forward projection procedure that is applicable to both libraries (point projection), we can observe that, in general, Halcon seems again to be a little better than OpenCV; making a quantitative comparison in this case would be significant, but the order of magnitude of the error (always <0.3mm with respect to the ground truth, and with the difference between the two libraries <0.05mm) is in both cases comparable to the accuracy of the ground truth (estimated around 0.15mm).

The real issue here is that we have reached a point where our assessment capability cannot provide any significant information: we are actually not assessing the accuracy of the libraries but that of our measurement setup (at a working distance of 1m an error of 0.3mm means an error lower than 0.02 degrees: but we have already noticed that a significant component of the error is related to the fact that our real world coordinates are expressed in an absolute reference system).
With OpenCV, of course, the two procedures based on forward projection are exactly equivalent because we use the OpenCV corner detector to localize verification points when we provide the image coordinates of verification points as input: we don’t get any additional information.

![Image0 (pix)](image)

Figure 2: Error according to backward projection on image 0  
(as a function of the number of calibration images)

The backward projection error related to image 0 (shown in Figure 2) is in general smaller than with other images; this is because these data are exactly the same that are used to perform calibration. In this case, however, OpenCV’s results are a little better, though the gap is so small that it looks negligible. Back projection on image 0 (in general, on calibration images) is the figure that is normally available when assessing calibration procedures: in our case this figure is also unaffected by possible movements of the test bed (which, of course, we tried to avoid completely) but the same argument as before applies when we consider its quantitative relevance.

These data show also that increasing the number of calibration images is not so important to improve the calibration quality. The very important thing, instead, seems to be that we could get a very good calibration with only 5 images because we guaranteed that in these images the calibration target covers the whole field of view of the camera.

The performance data that we have collected can be assembled in different ways to investigate the behavior of the two libraries with respect to specific parameters of the setup. As described in [10], in all cases we vary one of the considered parameters, and we average the results across all other parameters.

When we consider the relevance of the quality of the physical target, what we can observe is that a checkerboard target (as used by OpenCV) yields better results; a target that is a matrix of circles (as used by Halcon), instead, is more robust and probably more suitable if one cannot use a high quality target.

In fact, when we use the same calibration data the results provided by both toolkits are almost the same. This is an indication that the mathematical engines used by the two libraries are likely similar, or at least perform similarly. Using a high quality target, the error yielded by OpenCV is almost half of that of Halcon, while using a low quality, printed target, Halcon accuracy remains practically unchanged while OpenCV error increases more than 4 times. One should take into account that the checkerboard used by OpenCV has a higher number of calibration points (56) than the matrix of circles used by Halcon (49): our impression is that while this fact could affect in some way the results, it cannot explain by itself the higher calibration accuracy we can achieve using the checkerboard target.

Changing the resolution of the camera didn’t yield the expected results, independently from the library. In fact we notice that the error, in the image space, increases almost linearly with the number of pixel of the
sensor’s side while in the world space it remains practically unchanged (error in pixels are not reported because, as stated above, they cannot be considered as significant). In this working conditions, thus, increasing the camera resolution is practically useless. Apparently the working distance (~1m) doesn’t match the used optic and this results in a too large field of view (>600mm×600mm), which doesn’t allow to obtain more precise results.

<table>
<thead>
<tr>
<th>Camera resolution</th>
<th>OpenCV</th>
<th>Halcon</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Forward projection points (error in mm)</td>
<td>Forward projection point (error in mm)</td>
</tr>
<tr>
<td>0.3 Mp</td>
<td>0.288</td>
<td>0.213</td>
</tr>
<tr>
<td>1.4 Mp</td>
<td>0.290</td>
<td>0.216</td>
</tr>
<tr>
<td>2 Mp</td>
<td>0.292</td>
<td>0.216</td>
</tr>
<tr>
<td>5 Mp</td>
<td>0.276</td>
<td>0.284</td>
</tr>
</tbody>
</table>

Table 3: Error according to forward point projection procedure (as a function of camera resolution)

<table>
<thead>
<tr>
<th>Focal lengths</th>
<th>OpenCV</th>
<th>Halcon</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Forward projection points (error in mm)</td>
<td>Forward projection point (error in mm)</td>
</tr>
<tr>
<td>6.5mm</td>
<td>0.300</td>
<td>0.263</td>
</tr>
<tr>
<td>8mm</td>
<td>0.273</td>
<td>0.202</td>
</tr>
</tbody>
</table>

Table 4: Error according to forward point projection procedure (as a function of camera optic)

Also changing the optic of the camera produces almost no effect (Table 4). That’s because the two focal lengths (8 and 6.5 mm) are too similar to notice the difference at that working distance; but their choice and the choice of the working distance were forced by the physical limitations of the test bench, constrained by the measurement tools used to compute the ground truth, and by the dimension of the sensors of the cameras. In this case, however, Halcon seems to work a little better with a longer focal length and the difference between the two cases is a bit more visible (~0.06mm or 20-30% based on forward projection, but still smaller in absolute value than the accuracy of the ground truth). OpenCV, instead, yields exactly the same results in both cases, showing to be a little more insensitive to the characteristics of optic (a shorter optic causes higher distortion and allows a wider FOV).

Putting all data together, our test bed did not allow us to measure any significant difference between the two libraries, though Halcon data look consistently a little better than those of OpenCV.

To conclude this section, we have a couple of observations about the results obtained when using Matrox MIL. As already mentioned, the images acquired are not suitable for the algorithm implemented by this library but, though not considering here the specific numerical measurements, we can draw some interesting qualitative considerations. Of course, since it is a single-view calibration, it’s not possible to observe the different behavior when the number of calibration images increases.

When changing the quality of the checkerboard target, unlike OpenCV that uses the same kind of target, the error increases to a lesser extent in proportion to its absolute value. It seems that it’s not so sensitive to the imprecision introduced by a low quality calibration target. However, this might be due to the fact that
the errors introduced by the unsuitability of the calibration image is much higher than that introduced by the target.

MIL has a different behavior also when considering different types of camera resolution and optics. Actually, considering 0.3M, 1.4M and 2M cameras its behavior is the same as that of OpenCV and Halcon, but using a 5M camera reduces the error in the world space by more than half, differently from previous results.

A final observation about MIL is related to the effects of changing the camera optics. We can notice that with a 6.5mm the error is 3 times larger than with the 8mm optic. This can be explained considering the fact that a shorter optic has a larger FOV and, so, the calibration image covers a portion of the camera FOV even smaller than the one covered in a camera that uses a 8mm optic. Considering this, we expect that if we had used a suitable image to calibrate, the error would have been much smaller, comparable with the errors obtained by the other libraries.

### 3.2 2D object detection challenge in orthogonal view images

This challenge, and next ones alike, is actually split into two parts, depending on the fact that we are dealing with either textured or untextured objects. This distinction is because the algorithms for the handling of the two classes of objects may be substantially different. As already stated, in this document only results related to untextured objects are presented.

Another point that is common between this and the next challenges is that for each search image we are considering a set of 5 possible objects that may be present: each of them may be present zero or multiple times (an upper limit equal to 10 is specified in the solutions, while the actual maximum number of instances is equal to 6, and this may increase the FPR and thus cause a pessimistic evaluation of the library behavior), except for the case of images belonging to the single instance data subset where an object may be just present or not.

![ROC curves computed over the entire dataset - Low Texture - Single Instances](image)

**Figure 3:** ROC curves over the complete dataset for untextured, single instance object detection\(^1\) (orthogonal view)

---

\(^{1}\) FPR = False Positive Rate = False Positives / (False Positives + True Negatives)

TPR = True Positive Rate = True Positives / Ground Truth Positives
Figure 3 shows the performance data for the detection of a single instance of untextured objects. As already stated (see sec. 2.2) we can examine these data from two points of view: the first, more classical and more academic, compares the behavior of the different libraries (notice that OpenCV is not considered here because it doesn’t provide any suitable function for the detection of objects of this type, at least in the case we are considering here where scale variations are part of the challenge) based on the AUC metric, while the second considers only the leftmost part of the ROC curves, where false positives should be avoided. It is also worth pointing out that the importance in relevant practical scenarios to maximize detection ability (i.e. high TPR in the ROC space) while keeping working at nearly-zero FPR has been recently emphasized also in an academic object detection challenge organized within a major international conference in the field of robotics [14].

From the first standpoint VisionPro performs better than the others (the AUC values are reported in Figure 3 and in Table 5 under column “Overall AUC”/“Total”), but if we consider the part of the curves left of the plateau, i.e. where FPR is very low, we can notice that the performance of MIL is superior: it reaches its plateau for a value of FPR of 0.015, twice faster than VisionPro (which, though, has a significantly higher plateau, >0.9). When FPR=0.015 the performance of Halcon and VisionPro is still noticeably less than that of MIL.

Similar considerations apply also when we consider the same problem but with multiple instances of objects (Figure 4. The AUC values are reported in Figure 4 and in Table 6 under column “Overall AUC”/“Total”).

Having a large number of object instances in the search image makes detection more difficult, in particular increases the likelihood of false positives: in fact we see that all libraries reach their plateau for values of FPR that are an order of magnitude higher. The level of the plateau, instead, doesn’t change significantly.

The dataset of this challenge has been conceived in order to highlight the difficulties of object detection and to evaluate the robustness to nuisances: therefore object detection in this dataset is probably much more difficult than it would be in a real application setting.

Because the object localization challenge (see sec. 0) is related to localization and not to detection, its dataset was created so as to allow for a significantly easier detection, and in this sense its dataset may be considered more realistic for some industrial object detection applications (N.B. the same argument applies
to the two challenges of object detection and object localization under perspective distortion, see sec. 3.3 and 3.5 respectively).

Thus, if we consider object detection over the dataset of the untextured object localization challenge we see that the behavior of all libraries improves significantly (Figure 13, Figure 14 and Table 9, Table 10) and the plateau of the curves (with TPR ≥ 0.9) is normally reached for extremely low values of FPR (which means the highest detection performance is reached without false positives, which is the required behavior in most industrial applications). In this case the best performances, both in case of single instance and of multiple instances, are reached by VisionPro. The behavior of MIL and Halcon is very similar but when we consider single instance vs. multiple instances their relative performance changes (MIL is better for single instance, Halcon for multiple instances).

![Figure 5: ROC curves over the 0.3Mp-8mm camera dataset for untextured, single instance object detection (orthogonal view): left sparse, right dense](image)

![Figure 6: ROC curves over the 0.3 Mp-8 mm camera dataset for untextured, multiple instances object detection (orthogonal view): left sparse, right dense](image)
Because behavior at low FPR is so relevant we wanted to check whether our curves, plotted over the whole set of values of FPR, are dense enough to provide reliable info even in this range. To do this we ran a new set of tests setting the detection algorithms with high values of the “input score threshold”: we did this only for a specific camera setting (VGA-8mm). Figure 5 and Figure 6 compare the results of the original (left) and of the denser (right) plotting schemes for this configuration for FPR in the range 0..0.05.

First of all we notice that with this setting the best results are those of Halcon: our explanation for this is that the development dataset is based on VGA images, so that our solutions are somehow optimized for this setting: when we apply them to the general dataset Halcon has apparently more difficulties than the other libraries to adapt to different configurations.

![ROC curves](image)

Figure 7: ROC curves for the known polarity case over the 0.3 Mp-8 mm camera dataset for untextured, single instance object detection (orthogonal view)

![ROC curves](image)

Figure 8: ROC curves for the known polarity case over the 0.3 Mp-8 mm camera dataset for untextured, multiple instances object detection (orthogonal view)
Figure 9: ROC curves for the unknown polarity case over the 0.3 Mp-8 mm camera dataset for untextured, single instance object detection (orthogonal view)

Figure 10: ROC curves for the unknown polarity case over the 0.3 Mp-8 mm camera dataset for untextured, multiple instances object detection (orthogonal view)

The sparser plotting scheme tends to underestimate the performance of all libraries, especially for multiple instance images, but the overall picture doesn’t change much: thus, the zoom of the sparser curves provides a view that is enough reliable.

As stated in sec. 2.7, when we implemented the solutions for all libraries, we decided to allow the handling of variation of local polarity, in order to have generic and robust solutions, even though, in the majority of industrial applications, on-line acquired objects have the same polarity of their reference models. In order to verify how this parameter affects the detection in our dataset, we ran additional tests on the 0.3 Mp-8 mm camera dataset, with fixed polarity settings for all libraries.
The results of such tests are shown in Figure 7 and Figure 8, for single instance and multiple instances: they can be compared to those of Figure 9 and Figure 10 respectively; these last figures have been derived for the 0.3 Mp-8 mm camera dataset from those previously analyzed. The plateaux of all curves remain similar and are reached at very similar TPR levels. The most visible effect is the regularity of the VisionPro and MIL curves for very low FPR, that was lacking when polarity was not fixed: VisionPro, in particular, is the library that can make the best use of the constant polarity information whilst Halcon doesn’t seem to be able to gain anything from it.

The previous considerations apply to performance figures that are averaged over the complete dataset of this challenge, which includes images of different cameras and various degrees of synthetically introduced nuisances: however, we can look at performance data in a different way that highlights the effect of each of these parameters on the behavior of each library. This approach applies to this, but also to the next challenges [5].

Table 5 and Table 6 show the effects of nuisances and of resolution changes: w.r.t. nuisances, performance figures are averaged over the whole dataset of images with a graduated amount of nuisance added via synthetic image processing; based on raw data a more analytical comparison could be made, but this goes beyond the scope of this paper. Additionally, only the AUC metric is considered here, not the zero-FPR.

Column “AUC over regular images” is based on a limited subset of the dataset that has been used for Figure 3, Figure 4 and column “Overall AUC-Total”: this subset includes only real images, taken under good lighting conditions and with no addition of synthetic nuisances (these images are different from those where nuisances have been synthetically added). The “Overall AUC” dataset is partitioned also into subsets with different resolutions: each table shows the average performance for each level of resolution and the average across all levels (column “Total”). This applies both to the single instance and the multiple instance datasets, but unfortunately the object sets in the two cases are different. All this makes comparisons more difficult: it explains, for instance, why values in the “AUC over regular images” column may be worse than the corresponding values on the “Overall AUC” columns. These data are anyway significant when we compare the behavior of different libraries against the same dataset or the differential behavior of libraries between the same two datasets.

Looking at Table 5 VisionPro performs better than the other libraries, and improves its performance with the increase of camera resolution, moving from VGA to 2Mp cameras: if this rule doesn’t apply to 5Mp cameras is probably because these cameras are perceivably very noisy (in fact all libraries show irregular behaviors with these cameras). VisionPro is quite stable also with respect to addition of nuisances.

Also Halcon handles nuisances very well, while MIL doesn’t. Halcon and MIL show an oscillatory behavior with the increase of camera resolution and their performance drops significantly with the 5Mp camera: a reason for this has already been stated, but one may notice that the performance loss of VisionPro with 5Mp cameras is smaller.

<table>
<thead>
<tr>
<th>Library</th>
<th>AUC over regular images</th>
<th>Total</th>
<th>0.3 MP</th>
<th>1.4 MP</th>
<th>2 MP</th>
<th>5 MP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VisionPro</td>
<td>0.978</td>
<td>0.963</td>
<td>0.954</td>
<td>0.962</td>
<td>0.974</td>
<td>0.960</td>
</tr>
<tr>
<td>MIL</td>
<td>0.945</td>
<td>0.883</td>
<td>0.915</td>
<td>0.896</td>
<td>0.906</td>
<td>0.813</td>
</tr>
<tr>
<td>HALCON</td>
<td>0.886</td>
<td>0.914</td>
<td>0.945</td>
<td>0.941</td>
<td>0.947</td>
<td>0.843</td>
</tr>
</tbody>
</table>

Table 5: AUC over the complete dataset as a function of camera resolution, with and without nuisances (untextured objects, single instance)

Where a reasonable comparison can be made, Table 6 confirms the results derived from Table 5.

An interesting aspect, which is not highlighted by our benchmark but which was apparent during the implementation of the solutions of this challenge is the ease of use of VisionPro, whose functions are largely auto-adaptive w.r.t. the operating conditions, e.g. camera resolution.
<table>
<thead>
<tr>
<th>Library</th>
<th>AUC over regular images</th>
<th>Overall AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>0.3 MP</td>
</tr>
<tr>
<td>VisionPro</td>
<td>0.672</td>
<td>0.911</td>
</tr>
<tr>
<td>MIL</td>
<td>0.688</td>
<td>0.876</td>
</tr>
<tr>
<td>HALCON</td>
<td>0.695</td>
<td>0.839</td>
</tr>
</tbody>
</table>

Table 6: AUC over the complete dataset as a function of camera resolution, with and without nuisances (untextured objects, multiple instances)

### 3.3 2D object detection challenge under perspective distortion

The conclusions we drew in case of orthogonal view images don’t apply completely to the case of images with perspective distortion. In particular:

- Halcon and VisionPro provide specific functions to handle object detection under perspective distortion without image rectification. However, whilst the Halcon procedure was practically applicable, we found the VisionPro procedure so slow that it didn’t complete in reasonable time (≤10s) for any image of our dataset. Thus, for VisionPro we implemented our own procedure based on a rough detection of an object using the orthogonal view method of VisionPro with a low score (acceptance threshold = 0.4), which proved quite robust, and a following check over the related region of interest using the specific method for perspective distortion.

- MIL doesn’t provide a specific procedure for object detection under perspective distortion without image rectification, so we used the same detection procedure we had used for the orthogonal view case.

Taking all this into account, Figure 11 shows that also in this case, based on the AUC metric, VisionPro performs best (AUC values are reported also in Table 7 under column “Overall AUC”/"Total").

![ROC curves](image)

Figure 11: ROC curves over the complete dataset for untextured, single instance object detection (perspective distortion)
Figure 12: ROC curves over the complete dataset for untextured, multiple instances object detection (perspective distortion)

In case of detection of untextured objects, with multiple instances and under perspective distortion, we can apply the general considerations that we drew for orthogonal view (Figure 12. AUC values are reported also in Table 8 under column “Overall AUC”/“Total”). In this case, VisionPro shows the best behavior both under the AUC metric and for low FPR.

We have already noticed (sec. 2.3) that the definition of this challenge ignores the fact that the effects of perspective distortion could, in several real world application cases, be eliminated via calibration and image rectification: thus, the problem we are considering in this challenge could be reduced to that of section 3.2: this is probably the rationale for MIL not providing a specific function for this problem.

<table>
<thead>
<tr>
<th>Library</th>
<th>AUC over regular images</th>
<th>Overall AUC</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td>0.3 MP</td>
<td>1.4 MP</td>
</tr>
<tr>
<td>VisionPro</td>
<td>0.939</td>
<td>0.771</td>
<td>0.779</td>
<td>0.768</td>
</tr>
<tr>
<td>MIL</td>
<td>0.868</td>
<td>0.507</td>
<td>0.532</td>
<td>0.485</td>
</tr>
<tr>
<td>HALCON</td>
<td>0.897</td>
<td>0.715</td>
<td>0.719</td>
<td>0.721</td>
</tr>
</tbody>
</table>

Table 7: AUC over the complete dataset as a function of camera resolution, with and without nuisances (untextured objects, single instance)

<table>
<thead>
<tr>
<th>Library</th>
<th>AUC over regular images</th>
<th>Overall AUC</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td>0.3 MP</td>
<td>1.4 MP</td>
</tr>
<tr>
<td>VisionPro</td>
<td>0.675</td>
<td>0.616</td>
<td>0.631</td>
<td>0.621</td>
</tr>
<tr>
<td>MIL</td>
<td>0.591</td>
<td>0.386</td>
<td>0.431</td>
<td>0.357</td>
</tr>
<tr>
<td>HALCON</td>
<td>0.609</td>
<td>0.560</td>
<td>0.565</td>
<td>0.576</td>
</tr>
</tbody>
</table>

Table 8: AUC over the complete dataset as a function of camera resolution, with and without nuisances (untextured objects, multiple instances)
As in the previous section, we conclude looking at the effects of nuisances and of resolution changes (Table 5 and Table 8): the two tables show that Halcon slightly improves its performance by increasing the sensor resolution, while VisionPro and MIL seem to get worse.

### 3.4 2D object localization challenge in orthogonal view images

![ROC curves](image)

Figure 13: ROC curves over the complete dataset for untextured, single instance object localization (orthogonal view)

All functions that implement object detection yield also localization information about the objects that have been detected. In the case of the detection challenges we used this localization information only as a discrimination criterion to identify false positives. Here, instead, localization information becomes the main metric of the challenge.

<table>
<thead>
<tr>
<th></th>
<th>VisionPro</th>
<th>MIL</th>
<th>Halcon</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSE in x-position</td>
<td>0,95</td>
<td>0,83</td>
<td>1,01</td>
</tr>
<tr>
<td>RMSE in y-position</td>
<td>0,93</td>
<td>0,79</td>
<td>1,00</td>
</tr>
<tr>
<td>RMSE in rotation</td>
<td>0,12</td>
<td>0,17</td>
<td>0,11</td>
</tr>
<tr>
<td>RMSE in scale</td>
<td>0,008</td>
<td>0,012</td>
<td>0,006</td>
</tr>
<tr>
<td>Average Accuracy in x-position</td>
<td>0,97</td>
<td>1,11</td>
<td>0,91</td>
</tr>
<tr>
<td>Average Accuracy in y-position</td>
<td>0,98</td>
<td>1,10</td>
<td>0,90</td>
</tr>
<tr>
<td>Average Precision</td>
<td>0,05</td>
<td>0,49</td>
<td>0,36</td>
</tr>
<tr>
<td>Area Under Curve</td>
<td>1,00</td>
<td>1,00</td>
<td>0,99</td>
</tr>
</tbody>
</table>

Table 9: Accuracy of untextured object localization in the image space: orthogonal view, single instance

On the other hand, because the challenge is related to localization, the dataset has been created so as to allow an easier detection than in the case of the detection specific challenges. In a sense, as already

---

2 In this and in related tables RMSE (Root Mean Square Error) of x- and y-position is in pixel, RMSE of rotation is in degrees, accuracy of x- and y-positions and precision are in pixel, RMSE scale is a-dimensional. All these quantities are described in detail in [15] and [15].
observed, this also provides a dataset for detection challenges that may be considered more realistic for some industrial application, and in fact these detection figures have already been commented (see sec. 3.2).

Table 9 and Table 10 show that, for untextured objects, the most accurate library is Halcon (see parameter Average Accuracy in x/y position based on the localization of the bounding box) and the most precise is VisionPro. Halcon achieves this highest accuracy thanks to its handling of rotation and scale, as shown by the first four rows of the tables: in practical application where scaling may be less important its advantage could be lower than it appears here (as we have already noticed in sec. 2.3 the emphasis that our challenges put on scale variations is perhaps excessive in a benchmark that focuses on current industrial applications).

Table 10: Accuracy of untextured object localization in the image space: orthogonal view, multiple instances

<table>
<thead>
<tr>
<th></th>
<th>VisionPro</th>
<th>MIL</th>
<th>Halcon</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSE in x-position</td>
<td>0.91</td>
<td>2.07</td>
<td>1.00</td>
</tr>
<tr>
<td>RMSE in y-position</td>
<td>0.90</td>
<td>1.19</td>
<td>1.00</td>
</tr>
<tr>
<td>RMSE in rotation</td>
<td>0.12</td>
<td>0.23</td>
<td>0.09</td>
</tr>
<tr>
<td>RMSE in scale</td>
<td>0.008</td>
<td>0.011</td>
<td>0.005</td>
</tr>
<tr>
<td>Average Accuracy in x-position</td>
<td>0.96</td>
<td>1.13</td>
<td>0.87</td>
</tr>
<tr>
<td>Average Accuracy in y-position</td>
<td>0.95</td>
<td>1.08</td>
<td>0.87</td>
</tr>
<tr>
<td>Average Precision</td>
<td>0.05</td>
<td>0.69</td>
<td>0.43</td>
</tr>
<tr>
<td>Area Under Curve</td>
<td>1.00</td>
<td>0.98</td>
<td>0.99</td>
</tr>
</tbody>
</table>

For the definition of terms accuracy and precision see also [8]. Accuracy: closeness of agreement between a measured quantity value and a true quantity value of a measurand. Precision: closeness of agreement between indications or measured quantity values obtained by replicate measurements on the same or similar objects under specified conditions.
MIL is apparently quite good at estimating x and y translation (first two rows of the tables, based on the localization of the centroid of the objects), but it is weaker with rotation and scale. From the standpoint of nowadays industrial applications, where scale changes are less relevant, MIL could actually behave better than it appears here.

In the specific case of images with multiple instances of objects the overall rank of the three libraries remains the same. Halcon and VisionPro actually seem to behave even better, except for the precision of Halcon that decreases. But the differences are larger for MIL, whose precision and whose capability of estimating x and y translation is almost twice as worse; also the RMSE of rotation worsens significantly: looking at the details, one can actually see that this worsening takes place for high resolution cameras, whilst for other cameras performance data are much more similar to those of the single instance case. The library with the smallest RMSE for translations, in this case, is VisionPro.

All data shown in Table 9 and Table 10 refer to an average behavior over all regular images that have no nuisances and over all camera resolutions. Of course, all measurements were performed exclusively on cases related to true positives. However, the discrimination criterion to screen out false positives is very tolerant: it requires that the difference between the returned x and y coordinates of an object and the corresponding ground truth must be more than 1/10 of the image width (an analogous criterion is used also in the orthogonal detection challenge). As a consequence these data, that may include cases that might actually be considered false positives under a more stringent criterion, are to be interpreted as pessimistic results. If a stricter discrimination criterion had been used (for instance a difference of only few pixels), the outcome could have been different: the performance in terms of ROC curves would have been worse, but the localization uncertainty measures would have improved.

3.5 2D object localization challenge under perspective distortion

The only library that fully supports 2D object localization under perspective distortion without image rectification is Halcon: MIL doesn’t provide a specific algorithm for object detection under perspective distortion and, strangely enough, even though VisionPro has a specific procedure for object detection under perspective distortion, this procedure doesn’t return the perspective transformation under which the object has been located: this information is available only if one uses CVL, another toolkit provided by Cognex. OpenCV handles only the detection of textured objects.

<table>
<thead>
<tr>
<th></th>
<th>Halcon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Accuracy in x-position</td>
<td>7,14</td>
</tr>
<tr>
<td>Average Accuracy in y-position</td>
<td>6,93</td>
</tr>
<tr>
<td>Average Precision</td>
<td>11,99</td>
</tr>
<tr>
<td>Area Under Curve</td>
<td>0,79</td>
</tr>
</tbody>
</table>

Table 11: Accuracy of untextured object localization in the image space: perspective distortion, single instance

What we can do here, for untextured objects, is therefore only to compare the performance of Halcon in case of object localization for orthogonal view images and for images under perspective distortion. As expected (compare Table 9 and Table 11, and Table 10 and Table 12) accuracy and precision, in this case, are much worse (an order of magnitude in case of single instance images, two orders in case of multiple instances) than with orthogonal view.

As in the case of orthogonal view, these data concern regular images without nuisances, for all camera resolutions, and only true positives were taken into account to carry out measurements: also in this case a similar argument applies to the one that has been discussed about Table 9 and Table 10 at the end of sec. 0; here, however, we may expect that the effect of undetected false positives could be significantly higher. An accuracy of ~70 pixels as it is shown in Table 12 would probably be considered an index of a false
positive in an orthogonal view image, but one must also think that, because of the way the dataset is created, in case of multiple instances the average perspective deformation is larger than in case of single instance images (see last point of sec. 3.3) and consequently the errors are expected to be larger.

Figure 15: ROC curves over the complete dataset for untextured, single instance object localization (perspective distortion)

Figure 16: ROC curves over the complete dataset for untextured, multiple instances object localization (perspective distortion)
<table>
<thead>
<tr>
<th></th>
<th>Halcon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Accuracy in x-position</td>
<td>79,60</td>
</tr>
<tr>
<td>Average Accuracy in y-position</td>
<td>68,29</td>
</tr>
<tr>
<td>Average Precision</td>
<td>64,32</td>
</tr>
<tr>
<td>Area Under Curve</td>
<td>0,38</td>
</tr>
</tbody>
</table>

Table 12: Accuracy of untextured object localization in the image space: perspective distortion, multiple instances

4 Conclusions

Section 2 has highlighted several problems related to our approach and to specific technical issues about the definition of metrics, challenges, ground truth. We are learning from errors and we have tried and we will try in the future to avoid most of the mistakes we have made; but there is a general point that should be made: probably, we could have avoided them if we had been supported in our effort by the active collaboration of the machine vision community at large, and specifically of vision library vendors. Their experience is invaluable, but unfortunately they have underestimated the possibility of advancing their algorithms through the availability of well-defined challenge tasks with ground truth and evaluation metrics.

In section 3 we have reported some preliminary results of our benchmark: due the all limits that we have highlighted, we should consider them cautiously. However, looking at these data we can draw here some general conclusions.

With respect to the detection task, in configurations that are more similar to those that one may expect to find in reasonable industrial applications (e.g. untextured objects and orthogonal view: when the view is not orthogonal but the object lies on a known plane one may expect to rectify images based on calibration) all libraries yield good results. With the dataset that includes nuisance effects, i.e. the most challenging one, VisionPro reaches the highest TPR, MIL the lowest FPR (thus the highest TPR at very low FPR) while Halcon fits in the middle. With a different dataset, less challenging and hence perhaps more realistic with respect to the working conditions of standard nowadays applications, things change a bit: all libraries have very high TPR even at 0 FPR, with VisionPro showing the best performance for all values of FPR.

If we consider the localization task (orthogonal view), we cannot make any statement of general value: each library looks better from some particular perspective, but not from all.

When we consider perspective distortion for untextured objects, it is apparent that one cannot handle this working condition robustly if it doesn’t have a dedicated algorithm and if image rectification is not performed. MIL doesn’t provide such an algorithm, and the specific one provided by VisionPro is not viable: in our experience it could not handle the degrees of freedom of the benchmark (a skew of ±45° with a possible rotation of up to 360° in the image plane) in a reasonable time frame.

In this paper we haven’t presented any data related to time efficiency: this is because we don’t have yet a complete picture for all possible hardware configurations.

Even though VIALAB, as a research project funded by Regione Emilia-Romagna, will finish at the end of November 2012, we hope that we will be able to continue the activity of the benchmark in some other framework. There are several activities that we would like to complete or start.

First of all we would like to complete or revise the definition of some challenges. As we have already stated, one of our initial goals was to define a challenge of object localization in the real world (2.5D object localization). Based on our experience, 2D camera calibration should also be introduced. These 2 challenges
and camera calibration should be based on a dataset where the accuracy and precision of real world coordinates that are part of the ground truth are significantly higher.

The software tool for the generation of synthetic images with an associated ground truth should be re-engineered and made public.

The best way we can imagine to make our dataset largely available, is to provide a web access to our benchmark framework: in this way anyone could upload a library and a solution he has developed, and could run it against the dataset of the related challenge. Our benchmark framework would thus become a development environment for library vendors.

In order to encourage library vendors to take part in the benchmark, we could allow the use of our framework for development purpose subject to fact that the released version of the library will be benchmarked.

There are other ways we can help the VIALAB benchmark project to progress further. If this benchmark is perceived as relevant by the vendors, it is more likely that they will be willing to cooperate. This is why it is important that journals related to machine vision follow our activities as it happened in the past [2][4]: they could also sponsor the activity as a service provided to their readers.

Our experience has proved that the definition of a challenge is itself a challenge: beside the conceptual part of it, which is already difficult (as we have highlighted we have experienced several problems in our definition of metrics and challenges), the workload related to the creation of the dataset with associated ground truth and to the implementation of the evaluation framework and the OpenCV based example solution (live alone the implementation of the solution for all libraries) is huge. We cannot manage it by ourselves, unless at a very slow pace, and this is probably the reason why some of the vendors, though apparently willing to take part in the benchmark, have actually contributed very little. A solution would be to rely on the cooperation of independent developers and the academia, even in case library vendors change their attitude. Again, this is more likely to happen if the project makes headway and it is perceived as worthy and feasible: the role of trade and scientific press for this cannot be underestimated.

With all its limits, that we have tried to identify, and when possible to correct, we think that the VIALAB benchmark has been a valuable attempt to provide an important tool for the development and the assessment of machine vision libraries. We think also that a continuation and an improvement of this project is very useful and possible: we have tried to provide an outline of some ways this could be made.

5 References


4 All VIALAB documents except for [10] can be downloaded from the VIALAB web site