

Vers le dévracage 3D flexible et low-cost par robot bi-bras: les expériences des S.mart Toulouse et Besançon

M. Taïx, V. Cadenat
LAAS-CNRS
7, av du colonel Roche
31031 Toulouse
michel.taix@laas.fr
viviane.cadenat@laas.fr

S. Dembélé, C. Perrard
FEMTO-ST/AS2M
25 rue Alain Savary
25000 Besançon
soukalo.dembele@femto-st.fr
christophe.perrard@femto-st.fr

Résumé— Cet article s'intéresse au problème du dévracage 3D de pièces à l'aide de robots bi-bras. Plus précisément, il s'agit de développer une commande référencée vision, flexible et robuste, pour des tâches de pick-and-place de pièces de taille millimétrique ou centimétrique. Nous avons abordé les problèmes de détection et de localisation des objets, des robots et de commande afin d'obtenir la précision et la répétabilité souhaitée. S.mart Besançon a ainsi développé un dispositif expérimental centré sur le robot Yumi et une optique télécentrique, avant de calculer le modèle 3D d'un ensemble de composants de montre mécanique. S.mart Toulouse a proposé une stratégie de commande référencée vision permettant au robot Baxter de saisir une pièce détectée par la vision. Cette stratégie favorise la collaboration avec l'homme. Ces contributions ont été validées par des résultats expérimentaux obtenus sur les robots Yumi et Baxter.

Mots-clés— Dévracage, Cobotique, Interaction, Asservissement visuel, Perception, Modélisation 3D.

I. INTRODUCTION

Depuis quelques années au sein des lignes de production, les tâches de manipulation simples (de type pick-and-place) commencent à être confiées à des cobots [1]. En effet, ces derniers ne risquant pas de mettre en danger la vie des opérateurs, leur intégration au sein de la ligne ne requiert pas le développement d'infrastructures particulières (e.g., cages) et le coût global de la ligne est donc réduit. Cependant, l'intégration de ces cobots reste freinée par plusieurs obstacles. Le premier est lié à la faible précision et répétitivité de ces cobots, ceux-ci étant très sensibles aux modifications de l'environnement. D'autre part, leur flexibilité est limitée car, même si en pratique une tâche de pick-and-place est très générique, en pratique le cobot n'est pas capable d'apprendre facilement une nouvelle tâche qui ne différerait pourtant que de très peu d'une tâche qu'il connaîtrait déjà.

L'objectif de ce travail est de développer une commande référencée vision, flexible et robuste, pour des tâches de pick-and-place de pièces de petite (millimétrique) ou moyenne taille (centimétrique). On suppose que le pick est réalisé dans

une situation où les pièces sont dans un vrac (dévracage). L'enjeu est de déterminer les stratégies de détection/localisation de pièces, de localisation du robot et de commande permettant d'obtenir la précision et la répétabilité souhaitée. D'une part il s'agit de prendre en compte la taille et de la nature de la pièce à manipuler, d'où les deux cas d'étude envisagés. D'autre part, il s'agit de déterminer comment un opérateur non-roboticien pourrait facilement implanter une nouvelle tâche pick-and-place que le cobot pourrait rejouer de façon robuste.

Pour des raisons pratiques, l'étude se focalise sur des cobots manipulateurs à deux bras [2]. L'avantage d'un tel dispositif est de pouvoir utiliser un bras pour une perception active par vision (bras caméraman) et utiliser l'autre pour les manipulations de pièces. En effet, la perception active ajoutée conjointement à la manipulation est indispensable pour améliorer la précision lors de l'exécution des tâches.

II. BIBLIOGRAPHIE

Le dévracage est un problème connu et traité depuis longtemps dans l'industrie manufacturière, avec notamment l'utilisation de solutions ad hoc comme les bols vibrants. Ces solutions sont très efficaces dans biens de cas, notamment lorsque les pièces sont de formes simples et la cadence élevée, mais elles ne sont pas suffisamment flexibles, ce qui a conduit au développement de solutions robotiques. Aujourd'hui de nombreux constructeurs ou intégrateurs proposent une solution de dévracage. Malheureusement ces solutions industrielles sont chères, et de nouveaux besoins non industriels sont apparus, tels que le dévracage de types variés de produits dans les centres de distribution des sociétés de commerce telles que Amazon. La recherche en robotique s'est donc remise à investiguer le problème, et notamment depuis le concours de dévracage initié par Amazon, Amazon Robotic Challenge, depuis 2015[3], [4], [5], [7] [8].

Les publications sont surtout focalisées sur l'utilisation de caméras 3D permettant la mesure de pose 6D des objets ciblés. On trouve notamment l'utilisation de stereo-caméra avec des

objets plats [9], de scanner 3D avec des réseaux de neurones [10], de scanner avec l'algorithme ICP [11], de caméra low-cost comme la Kinect [8], [12]. Dans ces publications on trouve également l'utilisation de robots à double bras [6], [9], [13].

III. DEVRACAGE PAR ROBOT BAXTER

Nous avons effectué une saisie d'objet par asservissement visuel. Nous avons évalué la nécessité de l'interaction avec l'opérateur et défini les points à apprendre au préalable. Nous avons également considéré que cet apprentissage était réalisé par un utilisateur n'ayant pas de connaissance particulière en robotique.

a. PRESENTATION DU ROBOT BAXTER

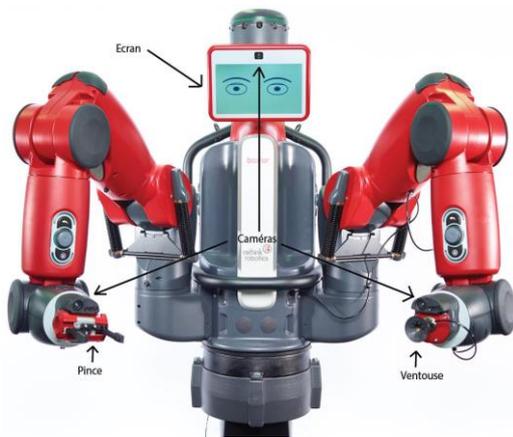


Figure 1. DISPOSITIF EXPERIMENTAL AVEC ROBOT BAXTER MONTRANT LES CAMERAS ET LA PINCE DANS LE BRAS DROIT.

Le robot Baxter (figure 1) a été créé par la société Rethink Robotics. Il s'agit d'un robot collaboratif possédant un buste relié à deux bras et une tête. Chaque bras est muni de sept liaisons, d'une caméra fixée sur chaque poignet et d'un organe terminal (pince ou ventouse). La tête est équipée d'un écran d'affichage et d'une caméra. Des boutons dédiés permettent d'interagir avec un opérateur. Une interface de programmation (API) permet de contrôler le robot via du code en python. L'environnement de travail choisi est ROS (Robot Operating System). Il s'agit d'un middleware offrant un ensemble d'outils, de bibliothèques et de conventions facilitant la programmation des robots [16]. Il propose un environnement de travail commun et pratique pour des roboticiens issus de tous horizons. Il permet de développer des programmes en C++ ou python. Nous effectuerons ici nos développements dans ce deuxième langage pour exploiter l'API évoquée plus haut.

b. RECONNAISSANCE D'OBJET

Les caméras du baxter sont très sensibles à la luminosité de l'environnement. De multiples tests ont en effet démontré que la reconnaissance de l'objet à saisir ne pouvait pas être réalisée à l'aide de la couleur et de la forme seules. Nous

avons donc opté pour une reconnaissance basée sur un AR-Tag, étant donné notre contexte de travail. Le package ROS choisi est ALVAR [16]. Ce package permet de : (i) détecter et de suivre des AR-tag; (ii) déterminer leur situation dans le repère caméra, et donc de calculer les points d'intérêt nécessaires à l'asservissement visuel (figure 2).

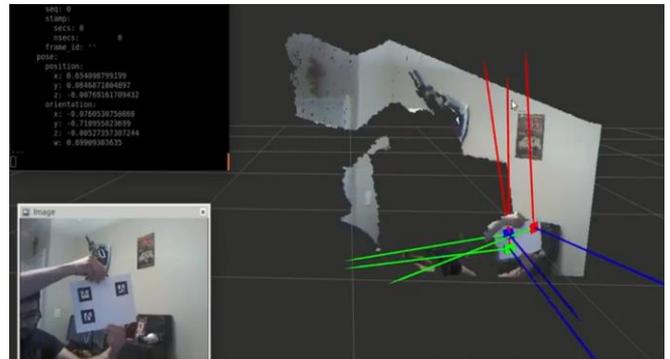


Figure 2. DETECTION AVEC ALVAR (GAUCHE : COORDONNEES DES POINTS & AR-TAG. DROITE : REPERES EXPRIMANT LA SITUATION DE CHAQUE POINT 3D DE L'OBJET).

c. ASSERVISSEMENT VISUEL

La commande référencée vision (ou asservissement visuel) est une structure de commande qui permet d'exploiter les informations visuelles issues d'une (ou plusieurs) caméra(s), embarquée(s) ou déportée(s) pour contrôler les mouvements d'un système robotisé [15]. Cette structure de commande recouvre deux grands types d'approches : l'asservissement visuel 3D et l'asservissement visuel 2D. Dans le premier, la tâche s'exprime en termes de situation à atteindre (pour la caméra ou pour l'objet) ; dans le second, elle est définie en fonction des indices visuels pertinents pour la tâche. Si la première approche présente l'avantage d'une bonne maîtrise des trajectoires effectuées par le robot, elle reste limitée par des problèmes de précision dus à l'extraction des indices visuels et à l'opération de reconstruction 3D nécessaire au calcul des informations requises pour la loi de commande. L'asservissement visuel 2D quant à lui ne souffre pas de cette limitation, même s'il est parfois plus compliqué de bien maîtriser les trajectoires du bras robotisé dans certaines conditions [15]. Afin de répondre au mieux aux exigences de notre cahier des charges en termes de précision, nous avons choisi de mettre en place un asservissement visuel 2D. Son principe est relativement simple. Il s'agit de réguler à zéro une erreur entre les indices visuels extraits de l'image courante notés ici s et ceux correspondant à une réalisation correcte de la tâche, s^* . Dans notre cas, s et s^* représentent respectivement les coordonnées en abscisse et en ordonnée courantes et désirées des quatre coins de l'objet dans le plan image préalablement déterminés par Alvar (cf. section III.B). L'objectif de l'asservissement visuel 2D est donc de réduire à zéro l'erreur $e = s - s^*$. Pour cela, l'approche la plus communément suivie consiste à imposer une décroissance

exponentielle [15] : $\frac{de}{dt} = -\lambda e, \lambda > 0$. En dérivant l'erreur, il est possible de montrer qu'une loi de commande en vitesse permettant de satisfaire cette dynamique est définie par [15] :

$$\frac{dq}{dt} = (LJ)$$

où q , L et J représentent respectivement la configuration du robot, la matrice d'interaction et la matrice jacobienne du bras robotisé. Le signe '+' désigne la pseudo-inverse de Moore-Penrose. J est calculée à l'aide de ROS. Quant à L , elle relie le mouvement des indices visuels dans l'image définie par la dérivée de s par rapport au temps au torseur cinématique de la caméra. Elle dépend généralement à la fois d'informations 2D et 3D [15]. Pour un point de coordonnées (x,y) dans l'image, elle est définie par [15] :

$$L_x = \begin{bmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix}$$

où Z désigne la profondeur du point considéré. Celle-ci peut être évaluée à chaque instant ou bien simplement calculée une fois pour toutes en $s=s^*$ [15]. C'est cette dernière solution qui a été choisie pour sa simplicité de mise en œuvre. Seule une valeur approchée de L sera donc calculée mais il est généralement admis que cela ne remet pas en cause la réalisation de la tâche [15]. Pour quatre points, la matrice d'interaction L sera donnée par la concaténation des quatre matrices L_x correspondantes.

Toutefois, l'asservissement visuel 2D ne peut être réalisé que lorsque les indices visuels sont disponibles. Or il peut arriver que durant la tâche ils ne soient plus calculables suite par exemple à un échec du traitement d'images ou une occultation temporaire de l'AR-tag. La stratégie de commande proposée a été étendue afin de résoudre ce problème. Tout d'abord, si les informations qui ont été extraites de l'image le permettent, les indices visuels manquants sont déduits à partir de ces dernières, d'un modèle 3D de l'objet et du modèle géométrique direct du robot. Sinon une stratégie plus élaborée a été mise en place. Elle consiste à exploiter le deuxième bras du Baxter pour en faire un "bras caméraman". L'idée est alors que ce dernier se positionne par rapport à l'AR-Tag en s'orientant vers lui pour avoir un angle de vue optimal tout au long de l'asservissement visuel. Ainsi, lorsque l'objet est perdu, il est possible de récupérer l'information de la situation actuelle pour continuer l'exécution de la tâche. Une commande complémentaire a donc été mise en œuvre sur le bras cameraman de manière à ce qu'il garde l'objet dans son champ de vue. En cas d'échec de ces deux solutions, l'asservissement visuel est temporairement arrêté et l'opérateur peut venir saisir le bras caméraman de manière à ce que l'objet soit à nouveau visible. L'interaction avec l'utilisateur est donc privilégiée dans ces travaux.

d. INTERACTION UTILISATEUR BAXTER

Afin de réaliser l'asservissement visuel il est nécessaire de donner la situation de prise de l'objet pour définir le s^* . Il

suffit de manipuler le Baxter dans la position finale et d'utiliser les boutons d'interface sur ses bras pour confirmer cette situation. L'opérateur utilise à nouveau les boutons d'interface pour définir la situation de dépose. Cette interface peut aussi aider l'asservissement du robot dans le cas de perte de l'objet avec les caméras. Il suffit alors que l'opérateur remette l'objet dans le champ de vue de la caméra.

Une fois les situations de prise et de dépose apprises, à l'aide de l'opérateur, plusieurs scénarios sont envisagés :

- Si l'environnement de dévissage est faiblement encombré, la détection de 'l'objet à saisir est facile et il suffit d'utiliser un seul bras pour détecter les amers visuels avec sa caméra et faire la prise de l'objet avec sa pince.
- Dans le cas d'un environnement plus contraint, le deuxième bras joue le rôle de caméraman et se positionne par rapport à l'AR-Tag en s'orientant vers lui pour avoir un angle de vue optimal tout au long de l'asservissement visuel. Ce bras caméraman est aussi intéressant lors de l'exécution de l'asservissement visuel car si l'AR-Tag est perdu de vue par le bras saisisseur, on ira récupérer l'information de la situation actuelle de l'objet et effectuera le changement de repère nécessaire entre les deux bras afin que l'asservissement visuel puisse se poursuivre.
- Dans tous les cas l'opérateur peut interagir avec le robot via les boutons d'interface.

e. RESULTATS EXPERIMENTAUX

Les résultats obtenus sont satisfaisants (figure 3). Cependant comme cela était attendu, le package Alvar n'est pas capable de détecter l'AR-tag en cas d'occlusion partielle de celui-ci. Or, lors des mouvements, il arrive que les pinces du robot masquent temporairement l'objet, entraînant un échec de l'asservissement visuel. L'utilisation du second bras vient corriger ce problème dans la majeure partie des cas. Toutefois, il est à noter que le bras caméraman n'étant pas directement au-dessus de l'amer et voyant donc celui-ci de côté, les poses renvoyées par Alvar sont plus imprécises que celles récupérées par la caméra du bras saisisseur. C'est pourquoi, elles ne sont utilisées que pour débloquer la situation quand l'objet est perdu. Dès que celui-ci redevient visible, la loi de commande est à nouveau nourrie à partir des indices visuels extraits du bras qui manipule. Cela se traduit généralement par une convergence de l'erreur vers zéro légèrement ralentie. La figure 4 montre l'évolution de l'erreur pour chaque point. Elle illustre bien la convergence de l'erreur vers zéro au cours de l'asservissement visuel. On notera sa petite augmentation sur la fin de la tâche. Cela correspond à une rotation de la caméra nécessaire pour corriger l'orientation finale.



Figure 3. POSITIONS INITIALE ET FINALE DES BRAS.

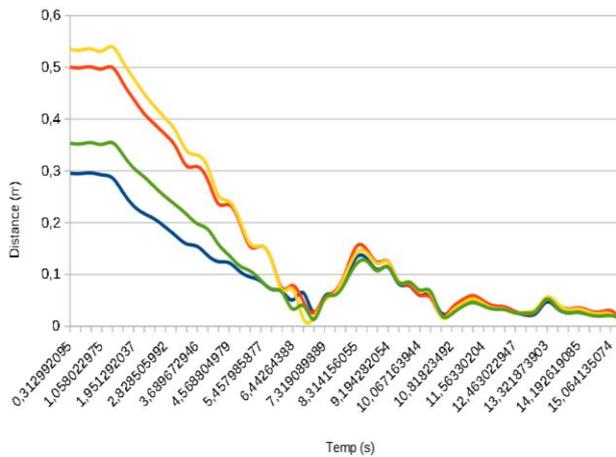


Figure 4. EVOLUTION DE L'ERREUR VISUELLE POUR LES 4 POINTS.

IV. DEVRACAGE PAR ROBOT YUMI

A. DISPOSITIF EXPERIMENTAL

Le robot Yumi conserve dans son bras gauche la pince de saisie habituelle, mais son bras droit est équipé d'une caméra couleur légère (46 g) à base d'un capteur 1/1.8'' pour 2560 x 2048 pixels en plein format (JAI 5000 C avec sortie USB 3.0). La caméra a été testée successivement avec un objectif standard de distance focale 25 mm, un objectif standard de distance focale 16 mm et un objectif télécentrique de distance focale 57.7 mm (CCS SE-110VT8 avec un grandissement x0.8, une distance de travail de 110 mm, une profondeur de champ de 2 mm). Elle est connectée à un PC sous Linux exécutant des applications en C++ avec les bibliothèques de traitement d'images OpenCV et de traitement de nuages de points PCL. La figure 5 montre une photo du dispositif.

La caméra est interfacée avec OpenCV à travers son kit de développement (SDK) CVB CameraSuite: les images au format CVB sont converties au format cv::Mat de OpenCV. La connexion entre robot et PC se fait par liaison TCP/IP. Les applications écrites en C++ du PC (incluant donc la vision) sont clientes d'un serveur écrit en Rapid et tournant sur le contrôleur du robot.

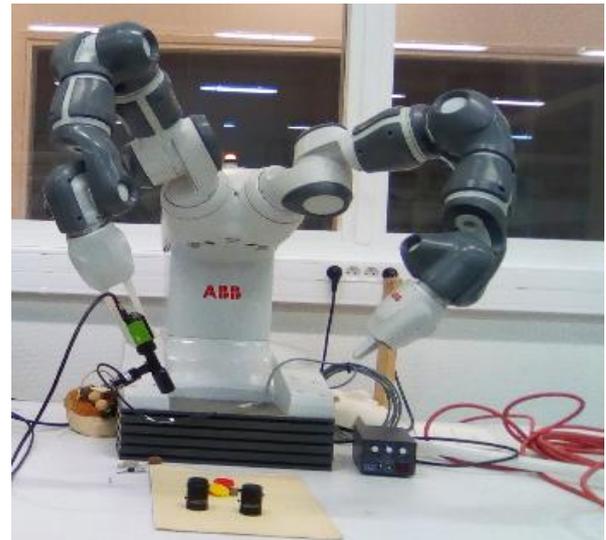


Figure 5. DISPOSITIF EXPERIMENTAL AVEC ROBOT ABB YUMI MONTRANT LA CAMERA DANS LE BRAS DROIT ET LA PINCE DANS LE BRAS GAUCHE.

Le projet est fondé sur un travail de thèse traitant de la reconstruction 3D auto-étalonnée pour l'imagerie en modèle affine [14]. Pour réutiliser ce travail il faut s'assurer que le modèle de l'imagerie (caméra avec optique) soit affine.

De manière générale, la modélisation géométrique d'une imagerie dépend de ses conditions d'utilisation, en particulier du rapport du champ de vision h à la distance de travail d : si $h/d \ll 1$, alors le modèle (ou caméra) est affine, dans le cas contraire il est perspective (ou central ou projectif).

Soit Q un point de l'espace et q son image sur le capteur. Les deux points vérifient:

$$\text{éq. (1): } q = K\Pi(R \vee t)Q$$

avec K la matrice des paramètres intrinsèques, la matrice de projection, R et t la matrice de rotation et le vecteur translation de la caméra par rapport au repère monde.

La matrice K comporte cinq paramètres (la distance focale en pixels selon les directions X et Y , le facteur d'orthogonalité du repère image, les coordonnées du foyer focal objet) pour le modèle perspective et trois pour le modèle affine (le foyer focal est situé à l'infini).

Avec des mires en damier adéquates et les fonctions OpenCV et PCL on a procédé à l'étalonnage des trois imageries: caméra avec optique 25 mm à une distance de travail de 60 cm, caméra avec optique 16 mm à une distance de travail de 60 cm et caméra avec optique télécentrique à une distance de travail 11 cm. Les erreurs de reprojection, c'est-à-dire la différence entre les positions réelles des points et leurs positions obtenues à partir du modèle, permettent de discriminer les deux modèles. Pour les optiques 25 mm et 16 mm les erreurs ont une grande plage de variation et une grande dispersion pour le modèle affine, et une moindre plage de variation avec une concentration autour de 0 pixel pour le modèle perspective: il en résulte que ces optiques sont décrites

par un modèle perspective et ne peuvent convenir pour les expériences (figure 6 pour l'optique 25 mm).

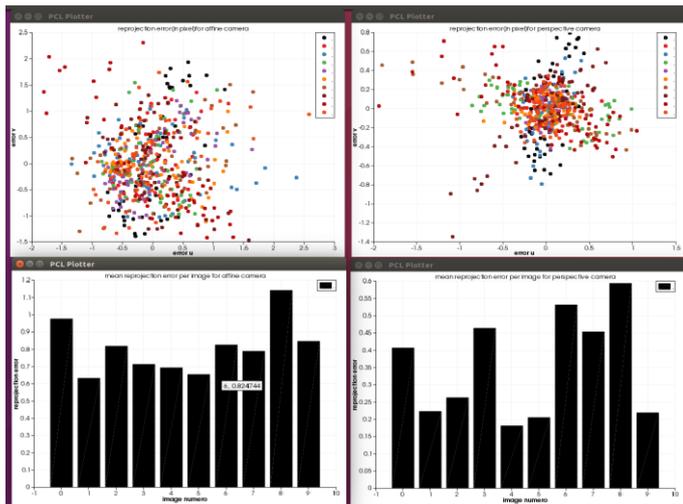


Figure 6. LA REPROJECTION DES ERREURS (HAUT) ET LES ERREURS MOYENNES PAR IMAGE UTILISEE (BAS) POUR L'OBJECTIF DE 25 MM

Pour l'optique télécentrique, pour les deux modèles, les erreurs sont faiblement dispersées autour de 0 pixel dans les mêmes proportions: comme le modèle affine comportent 2 degrés de liberté en moins que le modèle affine, on en déduit que l'optique télécentrique est bien affine (conformément du reste au principe de la télécentricité qui consiste en une annulation de l'effet perspective) (figure 7).

Pour la suite des expériences il faudrait donc utiliser la caméra avec l'optique télécentrique.

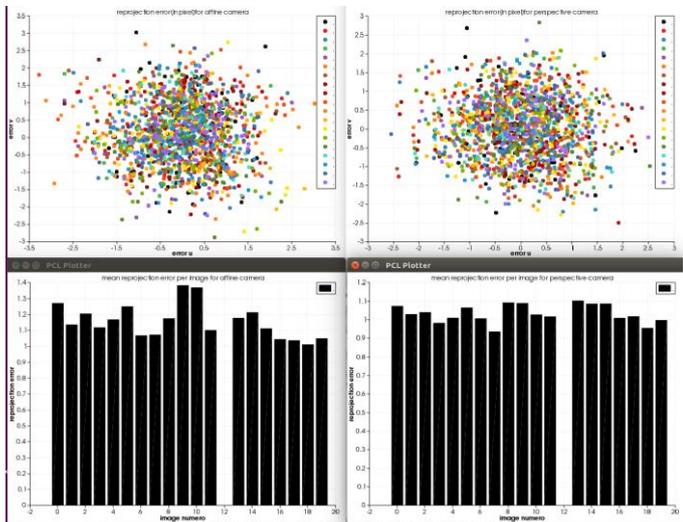


Figure 7. LA REPROJECTION DES ERREURS (HAUT) ET LES ERREURS MOYENNES PAR IMAGE UTILISEE (BAS) POUR L'OBJECTIF TELECENTRIQUE

C. RECONSTRUCTION 3D

Le calcul du modèle 3D (sous la forme d'un nuage de points 3D) de la scène comprend les étapes suivantes:

1. l'acquisition d'au moins trois images de la scène avec la caméra à optique télécentrique, à partir de trois positions du bras porteur du robot,
2. l'auto-étalonnage de la caméra pour déterminer le modèle et les mouvements de la caméra,
3. la rectification des images pour simplifier la recherche de correspondance entre paire d'images,
4. la mise en correspondance dense entre paire d'images, c'est-à-dire le calcul de tous les pixels communs entre paires d'images,
5. la triangulation pour déterminer le point 3D de chaque correspondance.

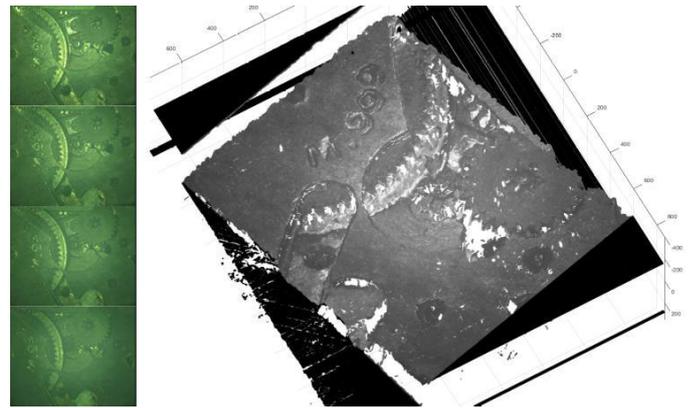


Figure 8. QUATRE IMAGES (GAUCHE) ET LA RECONSTRUCTION 3D RESULTANTE (DROITE)

L'auto-étalonnage de la caméra rend pratique la procédure de reconstruction, il n'est plus nécessaire d'utiliser une mire dédiée, les images de travail sont directement utilisées. Dans ce papier l'auto-étalonnage consiste à formuler une fonction coût régularisée ayant pour arguments les paramètres de la caméra (modèle affine et mouvement) et à calculer son optimum global grâce à une méthode d'optimisation non-linéaire avec contraintes de limites.

La figure 8 présente quatre photos d'une montre mécanique et le nuage de points obtenu confirmant la pertinence des développements.

V. CONCLUSION

Ce papier a présenté des travaux préliminaires en vue de réaliser le dévrage 3D avec des robots bi-bras.

S.mart Toulouse s'est intéressé à la prise de pièce reconfigurable référencée vision dans un contexte de robotique collaborative sur un robot Baxter. Ses contributions ont porté sur la détection de l'objet d'intérêt dans l'image et le pilotage du bras par commande référencée vision. La stratégie de commande a été développée de manière à intégrer l'utilisateur dans l'optique d'une collaboration homme/robot. Les résultats obtenus sont très satisfaisants et ont permis d'ouvrir différentes perspectives : améliorer la commande du bras caméraman pour maximiser la perception de l'objet, différencier les objets en leur attribuant un numéro détectable

avec Alvar pour pouvoir traiter le problème du vrac, exploiter la redondance offerte par les deux bras, traiter le problème de la reconfiguration.

S.mart Besançon a commencé par développer un dispositif expérimental centré sur le robot double bras ABB Yumi. Celui-ci comprend une caméra embarquée par le bras droit dudit robot et connecté à un PC qui communique avec le robot par liaison TCP/IP. Après avoir rendu fonctionnel le dispositif (interfaçage caméra avec OpenCV en C++, serveur Rapid), il a été démontré que le dispositif ne permet pas d'utiliser les optiques standard (la distance de travail ne permet d'avoir un modèle affine), au contraire de l'optique télécentrique qui donne bien un modèle affine. Les développements ont permis de calculer le modèle 3D d'une scène de composants de montre mécanique, validant les idées avancées. Cependant, les résultats ne sont que préliminaires, il reste encore beaucoup de travail à faire pour réaliser du dévracage. Les expériences ont été perturbées par l'apparition d'erreurs du côté du robot qu'il va falloir intégrer au serveur. Les temps de calcul du modèle 3D ne sont pas négligeables non plus et nécessitent un travail approfondi d'optimisation. A l'avenir il est envisagé d'utiliser une caméra Intel Realsense qui peut donner un flux de nuages de points à 30 Hz, permettant de se focaliser sur la reconnaissance des cibles et leur manipulation.

Grace à ce projet une véritable coopération a pu avoir lieu entre les deux sites S.mart. En dehors des échanges par vidéoconférence sur les problématiques académiques et industrielles liées au projet, nous avons organisé deux jours de travail sur Toulouse afin de montrer à nos partenaires l'intérêt de l'utilisation des outils autour de ROS.

VI. REMERCIEMENT

Ce travail a pu être réalisé grâce au financement par le réseau S.mart du projet *Devraflex* entre les sites S.mart de Besançon et de Toulouse.

VII. REFERENCES

[1] Haddadin S., Suppa M., Fuchs S., Bodenmüller T., Albu-Schäffer A., Hirzinger G. *Towards the Robotic Co-Worker*. Robotics Research. Springer Tracts in Advanced Robotics, vol 70. Springer, 2011.

[2] Christian Smith, Yiannis Karayiannidis, Lazaros Nalpantidis, Xavi Gratal, Peng Qi, Dimos V. Dimarogonas, Danica Kragic. *Dual arm manipulation—A survey*. Robotics and Autonomous Systems vol. 60, 2012.

[3] Nikolaus Correll, Kostas E. Bekris, Dmitry Berenson, Oliver Brock, Albert Causo, Kris Hauser, Kei Okada, Alberto Rodriguez, Joseph M. Romano and Peter R. Wurman. *Analysis and Observations from the First Amazon Picking Challenge*. IEEE Transaction on Automation Science and Engineering (T-ASE), 10.1109/TASE.2016.2600527, 2017.

[4] Clemens Eppner, Sebastian Höfer, Rico Jonschkowski, Roberto Martín-Martín, Arne Sieverling, Vincent Wall, Oliver Brock. *Lessons from the Amazon Picking Challenge: Four Aspects of Building Robotic Systems*. Robotics: Science and Systems, USA, June 18-22, 2016.

[5] Kensuke Harada, Weiwei Wan, Tokuo Tsuji, Kohei Kikuchi, Kazuyuki Nagata, and Hiromu Onda. *Initial Experiments on Learning-Based Randomized Bin-Picking Allowing Finger Contact with Neighboring Objects*. IEEE International Conference on Automation Science and Engineering (CASE), 2016.

[6] Wim Abbeloos and Toon Goedemé. *Point Pair Feature based Object Detection for Random Bin Picking*. ArXiv:1612.01288v1. 5 Dec 2016.

[7] Dirk Buchholz. *Bin-Picking: New approaches for a classical problem*. Springer. 2016

[8] Sangseung Kang, Kyekyung Kim, Jaeyeon Lee and Joongbae Kim. *Robotic vision system for random bin picking with dual-arm robots*. MATEC Web of Conferences 7 5 , 07003 (2016)

[9] Max Schwarz , Christian Lenz, Germán Martín García, Seongyong Koo, Arul Selvam Periyasamy, Michael Schreiber, and Sven Behnke. *Fast Object Learning and Dual-arm Coordination for Cluttered Stowing, Picking, and Packing*. Proc. of ICRA 2018.

[10] Wen-Chung Chang 1 · Chia-Hung Wu. *Eye-in-hand vision-based robotic bin-picking with active laser projection*. Int J Adv Manuf Technol (2016) 85:2873–2885

[11] Chintan Mishra, Zeeshan Ahmed Khan. *Development and Evaluation of a Kinect based Bin-Picking System*. Master of Thesis, University of Malardalen, 2015.

[12] Kensuke Harada, Weiwei Wan, Tokuo Tsuji, Kohei Kikuchi, Kazuyuki Nagata and Hiromu Onda. *Iterative Visual Recognition for Learning Based Randomized Bin-Picking*. ArXiv 1608.00334v1, Aug. 2016

[13] Andrey Kudryavtsev. *3D reconstruction in scanning electron microscope*. PhD thesis, Université de Franche-Comté, 2017.

[14] François Chaumette, S. Hutchinson. *Visual servo control, Part I: Basic approaches*. IEEE Robotics and Automation Magazine, 13 (4), 2006.

[15] Robot Operating System (ROS). *The Complete Reference* (Volume 3). Editors: Koubaa, Anis. Springer, 2018.