INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

CONCOURS EXTERNE Ingénieur de recherche

DIS 37

BAP E - INFORMATIQUE ET CALCUL SCIENTIFIQUE

Ouvert au titre de 2008
Arrêtés du 1er avril 2008 parus au JO du 5 avril 2008

EPREUVE ECRITE

25 juin 2008

Note sur 20 – Coefficient 3 - Durée : 3h00

Le candidat peut traiter les questions dans l'ordre de son choix.
Veillez à respecter l'anonymat dans les réponses.

Aucun document autorisé. Ni calculette, ni ordinateur, ni accès Internet.

Une attention particulière sera portée à la qualité de la présentation, à la grammaire et à l'orthographe.

QUESTION 1 : étude de dossier (10 points)

AVANT-PROPOS

L'activité de recherche de l'INRIA s'appuie sur environ 150 équipes (appelées EPI, équipes-projets INRIA), de 10 à 30 personnes, réparties dans les 8 centres de l'institut (CRI). L'activité de développement technologique qui accompagne ces recherches est pilotée par une direction nationale, la D2T (Direction du Développement Technologique), qui coordonne également les services de support dédiés au développement dans les centres, les SED (Service d'expérimentation et de développement). Pour améliorer la qualité et la visibilité des développements à l'INRIA, la D2T soutient, sur appel à propositions interne, des « Actions de développement technologiques » (ADT) rassemblant des EPI et des SED en vue d'un projet de développement particulier d'une durée de 2 à 4 ans. Les ADT sélectionnées se voient doter de moyens spécifiques en ingénieurs, équipement et fonctionnement, et les résultats de l'ADT sont évalués en relation avec les objectifs.
Le sujet de la question 1 concerne l'étude d'une proposition d'ADT.

ETUDE DE DOSSIER D'ADT
Deux dossiers au choix sont proposés à l'analyse des candidats :
-SensTools, en anglais, concernant des outils logiciels et matériels pour des réseaux de capteurs sans fil
-VISP, en français, concernant une bibliothèque logicielle pour le contrôle de robots par caméras.
Le candidat choisira UN SEUL de ces dossiers et rédigera un rapport sur celui-ci, destiné à son responsable hiérarchique, selon le plan suivant :

1- Synthèse de la proposition (1 à 2 pages)
2- Evaluation de la proposition selon les critères indiqués dans l'appel à propositions (1 page)
3- Avis final et recommandations (1 page):
    a. pour la direction : faut-il ou non soutenir et pourquoi? si oui à quelle hauteur et dans quelles conditions ?
    b. pour les proposants : points éventuellement à clarifier/améliorer

DOCUMENTS JOINTS
- Appel ADT 2008
- Plan de demande d'ADT
- Proposition SensTools
- Proposition VISP

Question 2 (5 points)

Un changement de version logicielle sur une plateforme expérimentale est prévu au cours du mois de juin et vous en avez confié la charge à 2 personnes de votre service.
Alors que vous êtes en déplacement professionnel, l'absence non anticipée d'une semaine d'un de vos deux collaborateurs ne permet pas de mettre à disposition en temps et en heure la plate forme. Par conséquent, les expériences prévues pour être présentées lors d'une conférence ne peuvent pas être effectuées. Cette situation suscite un conflit grave avec les chercheurs concernés qui mettent en cause l'absence de réactivité de votre service.

- Comment gérez-vous cette situation tant avec les utilisateurs qu'avec votre collaborateur ?

- Début Juillet, ont lieu les entretiens annuels d'activité au sein de votre service. Le collaborateur mis en cause est éligible à une promotion. Comment conduirez-vous cet entretien ?

- Rédigez, en moins de 20 lignes, l'appréciation que vous porterez sur son dossier d'entretien annuel d'activités.

Question 3 (5 points)

Vous devez mettre en place une « forge ». L'objectif d'une forge est de permettre à plusieurs développeurs de participer ensemble au développement d'un ou plusieurs logiciels.

- Quels sont les outils élémentaires que doit contenir une Forge ?
Rappelez pour chaque outil, les fonctions qu'il réalise.

- Suggérez d'autres fonctions supplémentaires, comme une messagerie instantanée, des fonctions de sécurité, ... et pour chacune de ces fonctions supplémentaires précisez comment elles peuvent permettre de recommander et de déployer des bonnes pratiques de développement.

Note : la question ne porte pas sur une forge particulière mais sur les fonctions mise en oeuvre par une forge.

Proposition pour une Action de Développement Technologique

ViSP

Visual Servoing Platform

http://www.irisa.fr/lagadic/visp/visp.html

Fabien Spindler, Éric Marchand, François Chaumette
EPI Lagadic
http://www.irisa.fr/lagadic

28 mars 2008

Résumé

La librairie ViSP (pour “Visual Servoing Platform” ou “plate-forme pour l’asservissement visuel”) développée dans l’EPI Lagadic permet le prototypage rapide d’applications permettant de commander ou de simuler des robots équipés de caméras en utilisant les techniques d’asservissement visuel. Une première ODL obtenue fin 2006, nous a permis d’améliorer la qualité de cette librairie, de la rendre multi plates-formes et de la diffuser sur la forge de l’Inria sous forme de logiciel libre. Les efforts menés jusqu’à présent sont encourageants et demandent maintenant à être amplifiés par l’intermédiaire d’une ADT. Elle permettrait assurément d’augmenter les fonctionnalités de la librairie et de renforcer sa diffusion auprès de la communauté. L’ADT ViSP aura pour responsable scientifique Eric Marchand, CR1 Inria, et pour responsable technique Fabien Spindler, IR2 Inria.

Table des matières

1 Objectifs de l'ADT

Pour tous les domaines d'application en vision robotique, il est crucial de mettre à disposition du développeur ou de l'utilisateur une librairie permettant la construction modulaire et sûre de tâches d'asservissement visuel. Depuis plusieurs années, l'EPI Lagadic développe et enrichit une telle librairie appelée ViSP pour "Visual Servoing Platform" ou "plate-forme pour l'asservissement visuel". Cette plate-forme est au cœur de l'ensemble des développements réalisés au sein de l'EPI Lagadic (voir paragraphe 6.3). Elle est également déjà utilisée à l'extérieur de l'EPI par une communauté d'utilisateurs (voir paragraphe 6.2).

Deux objectifs sont mis en avant dans cette demande d'ADT :

1. Renforcer la diffusion et la pérennisation de la plate-forme logicielle ViSP en tant que logiciel libre.
2. Réaliser des développements supplémentaires significatifs visant à intégrer les travaux de recherche récents de l'EPI Lagadic et à proposer à la communauté un outil plus performant comportant les dernières avancées dans ce domaine.

La réalisation de ces objectifs doit permettre :

- de pérenniser et de mieux valoriser les travaux de recherche de l'EPI Lagadic pour accentuer sa visibilité,
- de faciliter les développements dans le domaine de l'asservissement visuel, aussi bien pour poursuivre des activités de recherche, que pour en permettre une meilleure diffusion dans le monde académique et industriel,
- de mutualiser les développements réalisés en asservissement visuel au sein de la communauté de recherche au niveau international,
- et enfin, de fournir à la communauté un outil commun d'expérimentation et de validation.

Par rapport aux thèmes prioritaires de la politique scientifique de l'Inria définie dans le plan stratégique 2008-2012, cette demande d'ADT s'inscrit bien évidemment dans l'axe : *"Interaction avec des mondes réels et virtuels"* où la robotique tient une part importante. L'utilisation possible de ViSP dans le contexte de la réalité augmentée relie aussi cette demande à cet axe prioritaire.

Pour réaliser ces objectifs, nous demandons 48 hommes par mois pour renforcer l'équipe assurant le développement et le support de la plate-forme logicielle ViSP. Notre préférence irait vers le recrutement d'un seul ingénieur durant les quatre prochaines années. De leur côté, l'EPI Lagadic et le SED de Rennes mettront tout en œuvre pour assurer le succès de l'ADT auprès d'une communauté la plus large possible en affectant des ressources permanentes (voir Tab. 3). Actuellement, l'équipe technique est composée de 1,20 hommes par mois. Elle sera ramenée à 0,20 hommes par mois en décembre 2008. C'est Fabien Spindler, IR Inria, qui assurera alors la pérennité de ViSP

Comme mentionné au paragraphe 2.3, ViSP est la seule contribution technologique sous la forme de logiciel libre permettant d'appliquer les techniques d'asservissement visuel tant sur des systèmes réels qu'en simulation. Cette plate-forme est devenue incontournable et indispensable à l'ensemble des membres de l'EPI Lagadic, mais également pour d'autres chercheurs situés dans d'autres laboratoires (voir section 6.2). La durée de vie de la plate-forme logicielle ViSP est directement liée à celle de l'EPI Lagadic. Aussi longtemps que Lagadic mènera des activités de recherche en asservissement visuel, ViSP sera à maintenir et à faire évoluer. Après l'ADT, la maintenance et la pérenité de ViSP seront assurés par des ressources permanentes ; Fabien Spindler pour ce qui concerne les points techniques et Eric Marchand pour les aspects scientifiques.

2 Contexte et état de l'art

2.1 Contexte scientifique

Les techniques d'asservissement visuel se situent à l'intersection des domaines de la robotique, de l'automatique et de la vision par ordinateur. Elles utilisent les informations fournies par un capteur de vision pour contrôler les mouvements d'un système dynamique. Ce système peut être réel dans le cadre de la robotique, ou bien virtuel pour l'animation d'entités artificielles ou la réalité augmentée. Ce thème de recherche est au cœur des activités de l'EPI Lagadic.

Quelle que soit la configuration du capteur, pouvant aller d'une caméra embarquée sur l'effecteur d'un robot à plusieurs caméras déportées, les recherches en asservissement visuel visent à sélectionner au mieux un ensemble d'informations visuelles fournies par le capteur. Il faut ensuite élaborer une loi de commande contrôlant les degrés de liberté souhaités, afin que ces informations atteignent une valeur désirée, qui définit une réalisation correcte de la tâche. On peut ainsi réaliser une grande variété de tâches de positionnement du système par rapport à son environnement, ou de poursuite d'objets mobiles, en contrôlant de un à l'ensemble des degrés de liberté du système. Cette approche permet de compenser les imprécisions des modèles, aussi bien du capteur que du système à commander.

Avec un capteur de vision, fournissant en première instance des informations 2D, la nature des informations visuelles potentielles est extrêmement riche. En effet, pour l'asservissement visuel, il est possible d'utiliser des informations 2D, telles que les coordonnées de points caractéristiques dans l'image, ou des informations 3D, fournies par un module de localisation exploitant les mesures 2D extraites. De cette richesse provient la problématique majeure de l'asservissement visuel, à savoir, parmi l'ensemble des informations potentielles, comment sélectionner celles qui fourniront un comportement satisfaisant au système.

Les travaux menés par l'EPI Lagadic sont avant tout de nature méthodologique, mais il est inconcevable en robotique de ne pas valider les résultats de recherche par des expérimentations sur des systèmes réels. Ainsi, plusieurs plates-formes matérielles de robotique permettent aux chercheurs de l'EPI de valider leurs travaux. La mise en œuvre des techniques d'asservissement visuel sur des systèmes robotiques réels nécessite également de développer des algorithmes de traitements d'images temps réel (c'est-à-dire à la cadence vidéo) pour extraire et suivre les mesures dans les séquences d'images acquises.

Depuis de très nombreuses années, les recherches menées en asservissement visuel sont fructueuses (la liste exhaustive des publications de l'EPI sur le sujet est disponible sur le site web de l'équipe). Même si le spectre scientifique et applicatif qu'elles recouvrent est très large, les applications robotiques sont et restent les applications privilégiées de l'asservissement visuel. On peut ainsi considérer des applications :

- de positionnement de robots, de préhension, de manipulation et de suivi d'objets. Ce sont les principales tâches réalisables par asservissement visuel dans le domaine de la robotique manufacturière ou d'intervention (nucléaire [22, 26], spatial [15], sous-marin [29, 30, 11], médical [20], robotique mobile [16], engins aériens [3], ...).

Cependant de nouvelles applications, très différentes, sont récemment apparues en dehors du domaine de la robotique :

- en vision par ordinateur où des asservissements visuels virtuels permettent la localisation [27, 6] ou l'estimation de mouvements 3D [40] ;
- en réalité augmentée (incrustation d'objets virtuels dans des images réelles) où le problème fondamental (le calcul de pose) peut également être formalisé par la réalisation

d'un asservissement visuel virtuel [6] ;
– en réalité virtuelle avec la commande des déplacements d'une caméra virtuelle dans un environnement virtuel. Les applications dans le domaine des jeux vidéo, de la cinématographie virtuelle et de la commande des mouvements d'avatars ou d'humanoïdes virtuels sont notamment envisageables [31] [10].

2.2 ViSP : une plate-forme logicielle pour l'asservissement visuel

Face à l'absence d'un environnement logiciel permettant le prototypage rapide de tâches d'asservissement visuel, nous avons développé la plate-forme logicielle ViSP (pour "*Visual Servoing Platform*") dont les principales fonctionnalités sont présentées figure 1. Son objectif est de permettre la conception d'applications portables et facilement adaptables à d'autres contextes de travail. Ainsi, grâce à ViSP, une application mettant en œuvre des techniques d'asservissement visuel peut d'abord être développée et validée en simulation ou sur une plate-forme matérielle particulière, avant d'être transférée et intégrée à moindres coûts sur la plate-forme matérielle d'un partenaire.

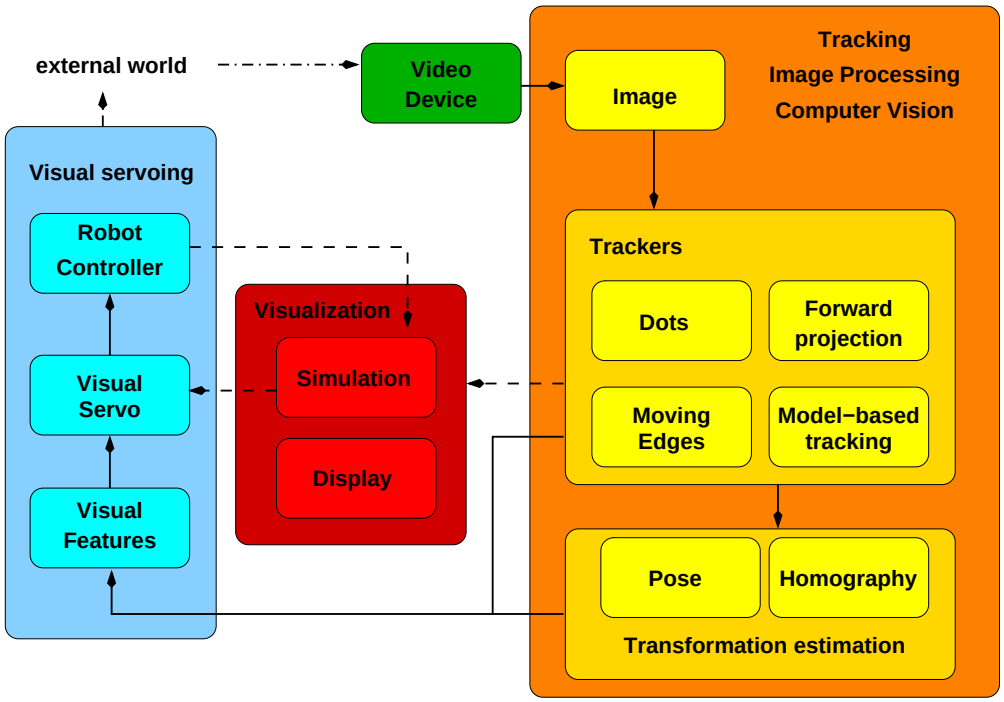


FIG. 1 – Fonctionnalités de la plate-forme logicielle ViSP dédiées aux applications exploitant les techniques d'asservissement visuel.

L'environnement réalisé permet la mise à disposition du programmeur d'un ensemble de briques élémentaires pouvant être combinées pour construire des applications plus complexes. Cet objectif est d'envergure en raison de l'emploi de matériels spécifiques (robot, carte d'acquisition d'images, etc.), mais aussi en raison de la très grande variété des applications potentielles, des lois de commande possibles, et des traitements d'images correspondants. ViSP présente les fonctionnalités requises pour ce type d'utilisation en mettant en avant une indépendance vis-à-vis du matériel, sa simplicité d'utilisation, son évolutivité et sa portabilité dans la plupart des environnements (Linux, Mac OSX, Windows).

ViSP permet le développement aisé de lois de commande d'asservissement visuel grâce à une vaste bibliothèque de tâches élémentaires de positionnement par rapport à des primitives visuelles variées (points, lignes, cercles, sphères, cylindres, etc.) pouvant être combinées pour la prise en compte d'objets plus complexes. Les aspects traitement d'images et vision par ordinateur sont aussi fondamentaux dans le contexte qui nous intéresse et de nombreux algorithmes (suivi d'objet, suivi de points d'intérêt, calcul de pose, estimation d'homographie, ...) sont aussi proposés. Précisons que ViSP peut aussi bien être utilisé pour la réalisation de tâches d'asservissement sur des robot réels que pour des prototypages rapides en simulation.

ViSP est écrit en C++ et utilise largement les fonctionnalités des langages orientés objets, en particulier pour assurer son évolution future et sa portabilité dans les environnements Linux, Mac OSX et Windows. Il est constitué actuellement d'environ 140 000 lignes de code (commentaires compris). Depuis 1999, les versions majeures de ViSP ont fait l'objet de dépôts APP.

2.2.1 Evolution de ViSP

Pour mieux comprendre l'évolution de la plate-forme, nous précisons ci-après quelques dates clés.

En mai 1999, la première version de la plate-forme logicielle se faisait connaître par l'intermédiaire d'une publication [24] dans une conférence internationale. A cette époque, la plate-forme était utilisée en interne par quelques rares initiés. Peu à peu, au sein de l'EPI Lagadic, de plus en plus d'utilisateurs ont vu le gain qu'ils pouvaient tirer de ViSP, à tel point que la plate-forme est devenue un "standard" interne.

En janvier 2005, fort de cette première expérience, les auteurs ont décidé de ré-écrire ViSP pour notamment améliorer son architecture et la rendre plus modulaire en exploitant davantage les capacités objets du C++. La figure 2 récapitule les dates importantes du logiciel depuis cette décision.

En juillet 2005, une première version *beta* a été diffusée sous licence QPL depuis le site web de l'EPI pour permettre à quelques personnes de la tester.

En décembre 2005, l'article *ViSP for Visual Servoing* [33] paru dans la revue *IEEE Robotics and Automation Magazine* à très large diffusion dans la communauté robotique assurait une "publicité" importante de cette nouvelle version du logiciel en décrivant en détail ses caractéristiques. Dans la foulée, une première version stable était mise à disposition pour la première fois sur le net en janvier 2006.

En avril 2006, la mise en œuvre de l'outil de configuration CMake nous a permis, d'une part, de rendre ViSP multi plates-formes, et d'autre part, d'automatiser les tests sur plusieurs cibles (OS et compilateurs différents). Des tests de compilation et d'exécution (voir Fig. 9), d'analyse de la mémoire (voir Fig. 10) et de couverture du code (voir Fig. 11) ont été déployés. Les tests de compilation permettent de détecter des erreurs de programmation. Les tests d'exécution nous permettent de vérifier le bon fonctionnement de certains modules et de comparer les résultats produits à ceux attendus. Les tests d'analyse de la mémoire nous permettent de détecter et de localiser les fuites mémoire. Enfin, les tests de couverture nous permettent d'identifier les portions du code non testées.

En juin 2006, ViSP a migré sur la forge de l'Inria pour bénéficier de l'ensemble de ses outils permettant de réaliser le développement collaboratif du logiciel. Cette importante étape avait aussi pour objectif de valoriser et de promouvoir la plate-forme auprès de la communauté en asservissement visuel tout en renforçant le savoir-faire de l'EPI Lagadic.

En décembre 2006, nous avons bénéficié d'une ODL visant à stabiliser et fiabiliser le cœur

du logiciel. L'accent a d'abord été mis sur l'amélioration de la qualité du logiciel et de sa documentation. De nombreux bogues ont été supprimés, de nouvelles fonctionnalités ont été introduites (outils de calibration de caméras, suivi de points d'intérêt), de nombreux exemples et programmes de test ont été développés. La documentation Doxygen du code est constamment améliorée et le site web dédié à ViSP enrichi. Deux versions stables du logiciel ont été diffusées en 2007, la première en février, la seconde en mai. Une nouvelle version stable va être diffusée début avril 2008.

Aujourd'hui, tous les membres de l'EPI Lagadic sont des utilisateurs de ViSP tout comme d'autres laboratoires tels le Lasmea à Clermont-Ferrand, le CEA à Fontenay aux Roses, le Joint Robotic Laboratory (JRL) au Japon ou l'Université Johns Hopkins aux Etats-Unis (voir paragraphe 6.2). Le nombre de téléchargements des versions stables (la dernière, ViSP-2.4.1 a été téléchargée 1005 fois, voir figure 5) et des publications associées ([33] a été téléchargée 1743 fois, voir figure 7) met en évidence l'intérêt de la communauté pour ce logiciel.

2.2.2 Indicateurs déjà mis en place

Pour avoir une vue d'ensemble de l'évolution de ViSP, nous pouvons comparer l'évolution de certains indicateurs en fonction des dates marquantes du projet (Fig. 2).

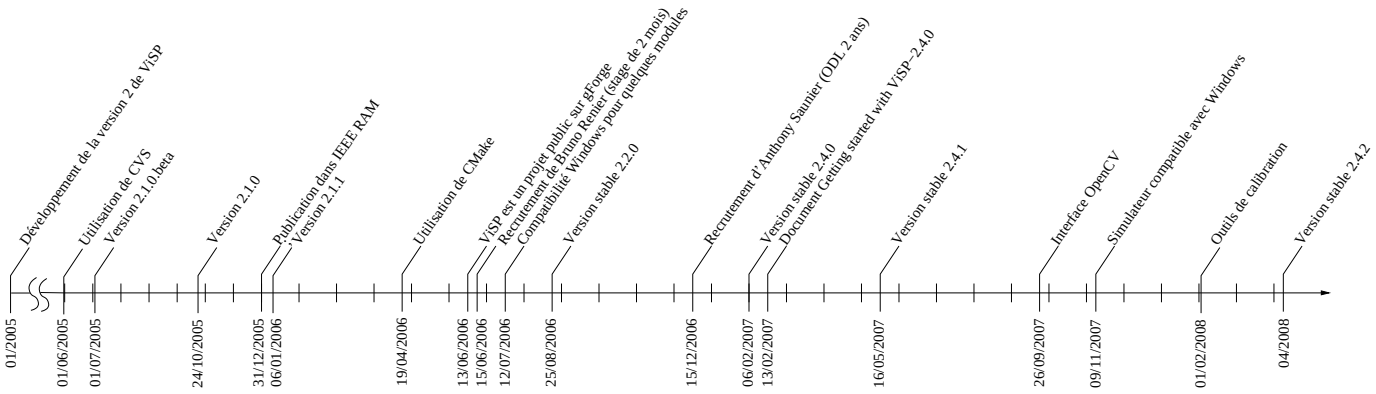


FIG. 2 – Dates marquantes de l'évolution de ViSP depuis la décision en janvier 2005 de développer la version 2 du logiciel, suivie de sa mise à disposition sous forme de logiciel libre à travers la publication [33] parue dans le journal IEEE RAM en décembre 2005. Depuis, ViSP est en constante évolution pour mieux répondre au besoin de ses utilisateurs. De gros efforts ont notamment été déployés pour porter ViSP sous Windows.

Nous pouvons aussi comparer ces évolutions d'une manière plus globale en fonction du nombre d'hommes par mois (HM) alloués au projet (voir Fig. 3) et constater logiquement que les périodes où le nombre d'hommes par mois est le plus fort correspond à des périodes de fortes augmentations du nombre de lignes de code dans ViSP (voir Fig. 4). Depuis janvier 2006, un certain nombre de développements ont été effectués sur la plate-forme logicielle ViSP. Nous avons une indication du volume du travail effectué en observant l'évolution du nombre de lignes de code (Fig. 4). En deux ans, le logiciel a doublé de taille (Fig. 4.a). Pour rendre compte du travail lié au côté multi plates-formes de ViSP, il est aussi intéressant d'observer l'évolution du nombre de lignes de code des fichiers de configuration (Fig. 4.b) liés d'abord à l'utilisation des Autotools, puis depuis avril 2006 à l'utilisation de CMake. Notons toutefois

que les graphes liés à l'évolution du nombre de lignes de code sont à analyser avec précaution ; la qualité d'un logiciel n'étant pas liée à sa taille.

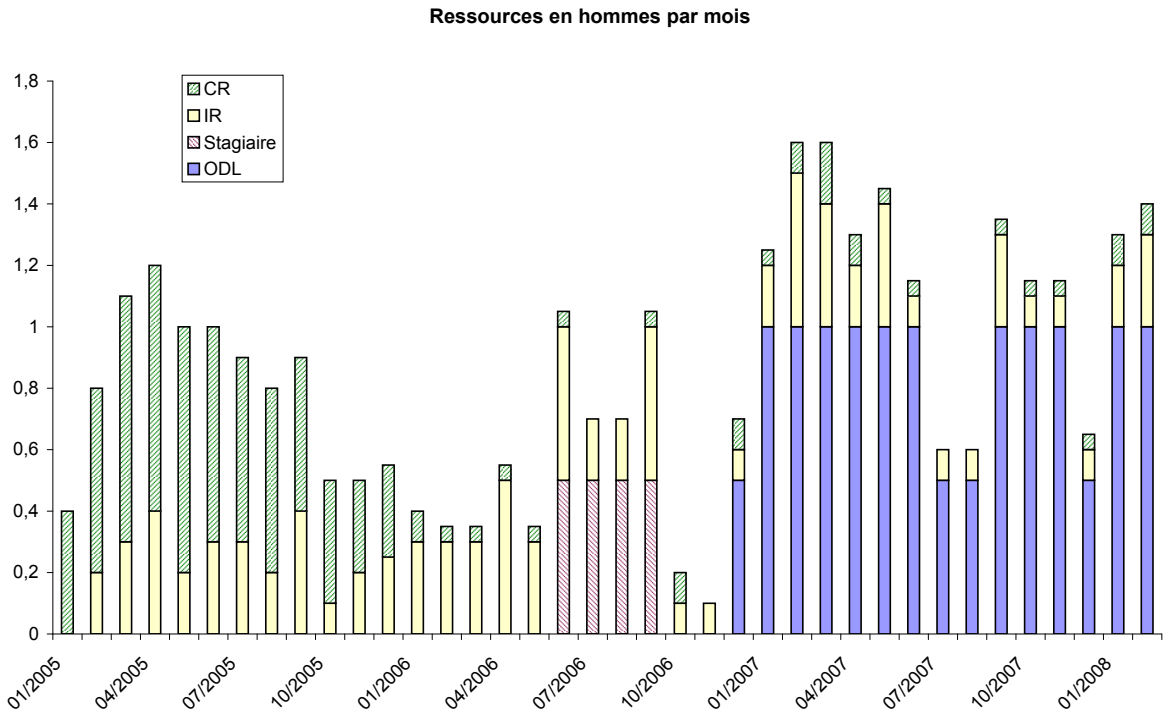


FIG. 3 – Evolution en fonction du temps, du nombre d'hommes par mois attribués à ViSP, depuis la décision prise en janvier 2005 de ré-écrire le logiciel pour développer la version 2 du logiciel.

Les différentes améliorations apportées à ViSP ont permis de diffuser trois versions stables de la librairie :

- ViSP-2.2.0 diffusée le 25 août 2006, téléchargée 251 fois depuis la forge de l'INRIA,
- ViSP-2.4.0 diffusée le 6 février 2007, téléchargée 473 fois,
- ViSP-2.4.1 diffusée le 16 mai 2007, téléchargée 1005 fois.

ViSP tient aujourd'hui le 25ème rang des projets les plus téléchargés sur la forge de l'Inria.

Une nouvelle version de ViSP (ViSP-2.4.2) sera diffusée début avril 2008. L'allure de l'évolution du nombre de téléchargements mensuel de ces versions (Fig. 5) souligne une montée en puissance dans l'utilisation du logiciel et montre que les utilisateurs sont demandeurs de versions récentes de ViSP.

Une version courante en cours de développement est également disponible sur la forge de l'INRIA et accessible via CVS. Le faible nombre de *checkout* effectués par mois (Fig. 6) laisse à penser que ce mode de téléchargement de la version la plus à jour de ViSP s'adresse plus à des spécialistes.

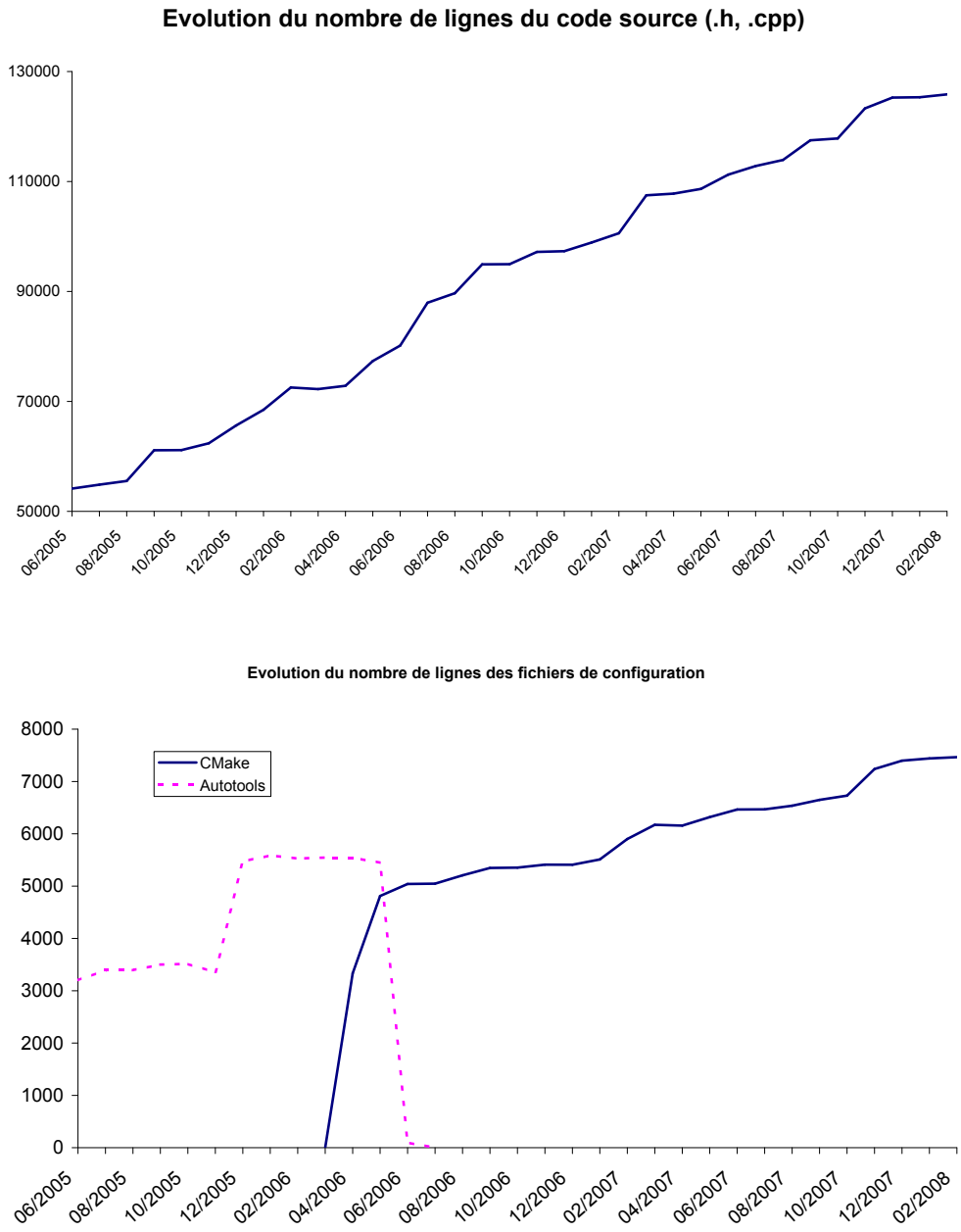


FIG. 4 – Evolution du nombre de lignes de code de ViSP depuis que le logiciel est géré par le gestionnaire de version CVS en juin 2005 : (a) Nombre de lignes du code source dans les fichiers *.cpp et *.h. Ces nombres incluent les commentaires, (b) nombre de lignes liées aux fichiers de configuration utilisés par Autotools et par CMake pour permettre la construction multi plates-formes du logiciel. Concernant les Autotools, il s'agit des fichiers de macros M4 (*.m4), et des fichiers Makefile.in permettant à Autoconf de produire les fichiers Makefile pour les cibles Unix. Concernant CMake, il s'agit des fichiers CMakeLists.txt et *.cmake. A l'origine du projet, les Autotools (notamment Autoconf) étaient utilisés pour générer les fichiers Makefiles permettant de compiler ViSP sur des OS du type Unix (Linux, MacOSX). Puis, peu à peu le besoin s'est fait sentir de porter ViSP sous Windows, environnement sur lequel les Autotools n'étaient pas utilisables. C'est pourquoi, à partir d'avril 2006, l'outil CMake a été introduit pour remplacer les Autotools. CMake permet de créer les fichiers de configuration compatibles avec tous les compilateurs supportés par les environnement Linux, MacOSX et Windows.

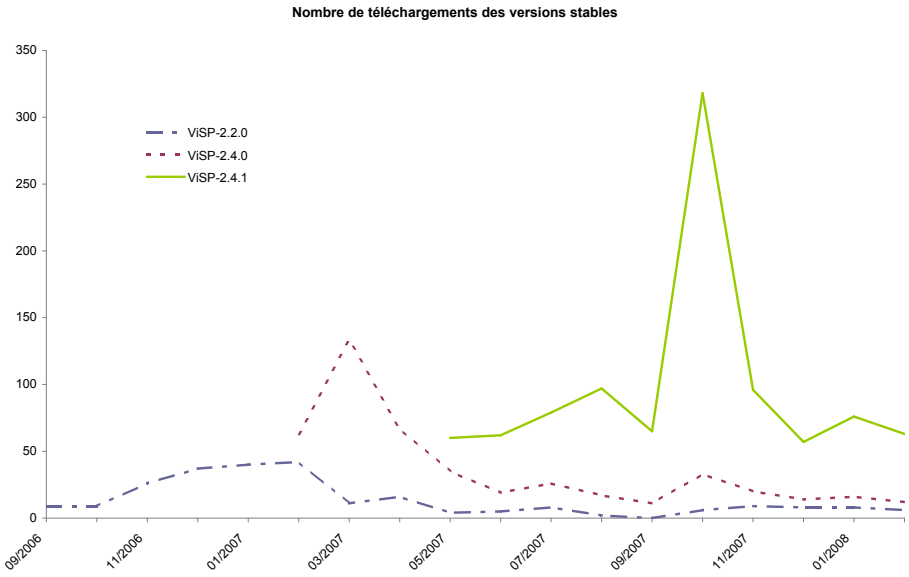


FIG. 5 – Évolution du nombre de téléchargements mensuel de ViSP depuis que le logiciel est un projet public sur la forge de l'Inria : ViSP-2.2.0 a été diffusée le 25 août 2006 et téléchargée 251 fois, ViSP-2.4.0 a été diffusée le 6 février 2007 et téléchargée 473 fois et enfin ViSP-2.4.1 a été diffusée le 16 mai 2007 et téléchargée 1005 fois. Ces données ont été obtenues à l'aide des outils de statistique de la forge.

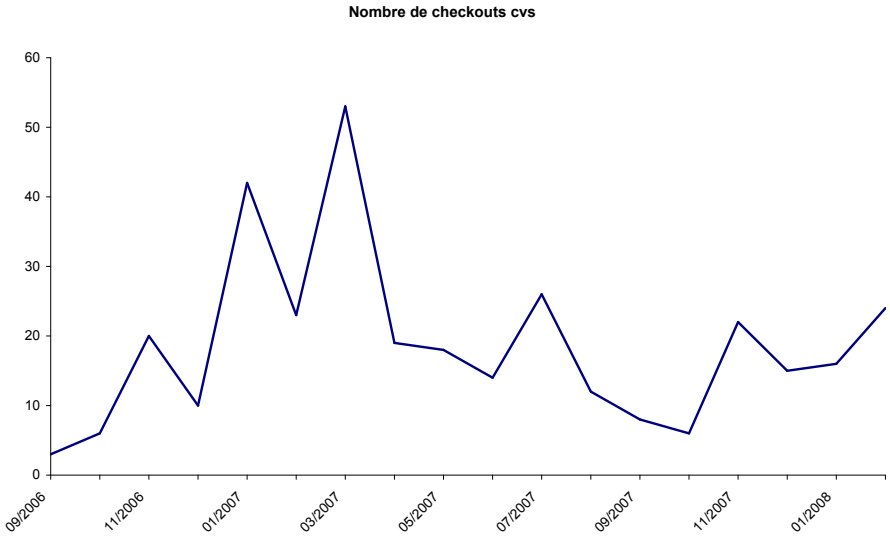


FIG. 6 – Évolution mensuelle du nombre de *checkout* CVS du code source réalisés depuis la forge INRIA.

Nous avons également observé le nombre de téléchargements depuis le site web de ViSP des publications [24, 33] et documentations [32, 42] relatives au logiciel. Les chiffres présentés figure 7 ont été obtenus grâce à l'outil de statistiques associé au web Lagadic.

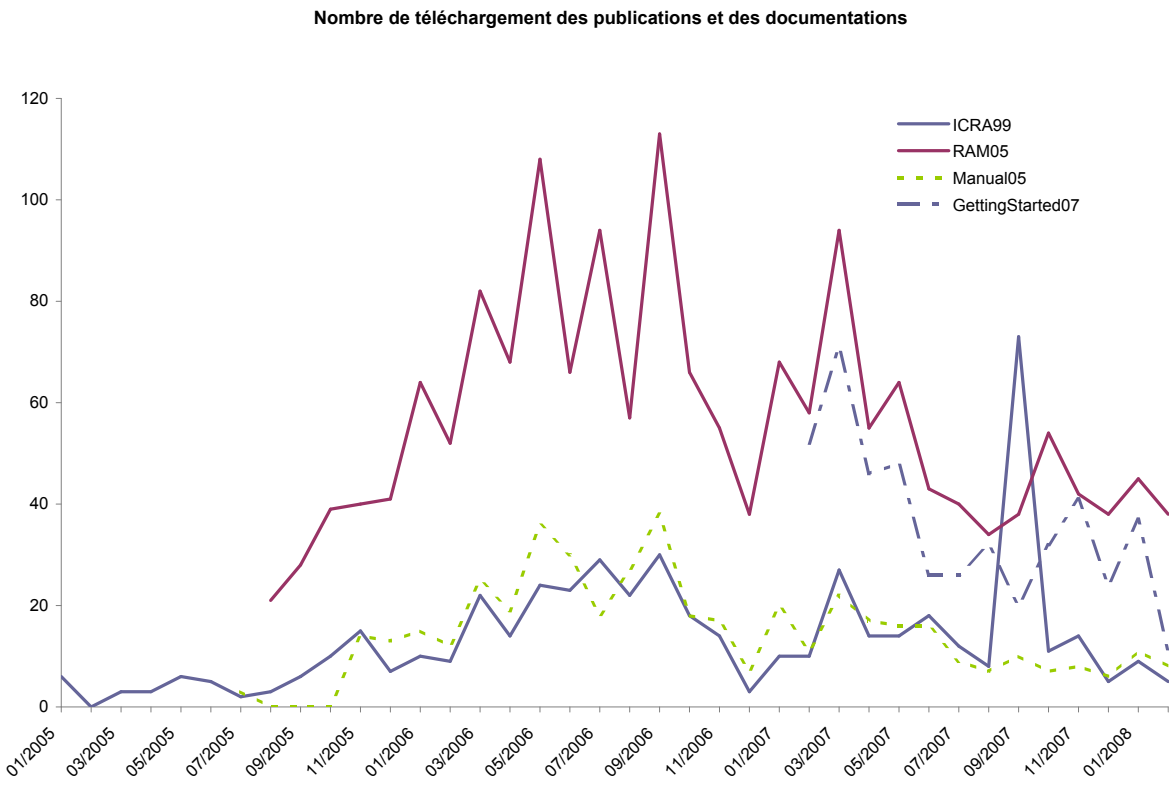


FIG. 7 – Évolution mensuelle du nombre de téléchargements des publications ICRA99 [24] (téléchargée 514 fois), RAM05 [33] (téléchargée 1743 fois) et des documentations Manual05 [32] (téléchargée 460 fois) et GettingStarted [42] (téléchargée 466 fois) relatives à ViSP. Ils s'agit du nombre de téléchargements réalisés depuis le site web de l'EPI Lagadic dont les statistiques sont connues. Ces chiffres ne tiennent pas compte d'autres sites web où il est possible d'obtenir certaines de ces publications (comme par exemple le site de IEEE). Notons par ailleurs que la publication RAM05 [33] a été acceptée en juillet 2005 et diffusée aussitôt sur le web de l'EPI Lagadic avant de paraître dans le journal en décembre 2005.

Les principaux pics (Figure 7) coïncident avec les dates des deux conférences majeures dans le domaine de la vision robotique : ICRA (International Conference on Robotics and Automation) et IROS (International Conference on Intelligent Robots and Systems).

- Pour ICRA : la soumission a lieu en septembre, la conférence en avril/mai de l'année suivante.
- Pour IROS : la soumission a lieu en avril, la conférence en octobre de la même année.

Ces chiffres sont incomplets puisqu'ils ne recensent que le nombre de téléchargements des publications depuis le site web de Lagadic. Les pdf (notamment des publications [24] [33]) sont

disponibles et téléchargeables depuis d'autres sites dont les statistiques ne sont pas connues.

2.3 Positionnement du développement

Il est impossible d'énumérer l'ensemble des laboratoires travaillant actuellement dans le domaine de l'asservissement visuel. Les principaux et les plus anciens sont situés aux États-Unis (Seth Hutchinson à Urbana-Champaign, Greg Hager à présent à Johns Hopkins), au Japon (Koichi Hashimoto à Tokyo) et en Australie (Peter Corke au CSIRO). En Europe et hormis en France, de nombreux laboratoires sont actifs depuis quelques années (le KTH à Stockhölm en Suède, l'Université La Sapienza à Rome, l'Université de Naples, l'Université de Sienne en Italie, l'Université d'Alicante et de Jaume en Espagne,...).

La communauté française est très active dans le domaine de l'asservissement visuel. Actuellement, les laboratoires suivants mènent des activités de recherche sur ce sujet : l'Inria Sophia-Antipolis, l'I3S, le Laas, le Lasmea, l'ISIR, le CEA, l'Ifremer, le LVR, le LIRMM, l'UTC et le LSIIT. Plusieurs projets nationaux du type ANR comportent une part non négligeable de recherches en asservissement visuel.

Le grand nombre de laboratoires travaillant dans notre domaine nous a conforté dans l'idée de proposer à la communauté un outil permettant de travailler sur une thématique de recherche importante.

A notre connaissance, il n'y a pas de systèmes concurrents ayant toute les caractéristiques de ViSP. La simulation de lois de commande d'asservissement visuel peut bien sûr être réalisée en utilisant des outils disponibles sous Matlab (eg, Matlab robotics Toolbox [7], visual servoing toolbox for MATLAB / Simulink [4]), mais ce genre de simulation se prête mal à une implémentation sur des robots réels. Les bibliothèques de vision par ordinateur et de traitement d'images sont plus nombreuses (XVision [18], The Epipolar Geometry Toolbox [34], The Machine Vision Toolbox [8], Intel Open Source Computer Vision Library (OpenCV), ...) mais elles ne sont en général pas adaptées aux problématiques de vision temps réel. Par ailleurs, elles ne contiennent aucun élément concernant les domaines de l'asservissement visuel et de la robotique.

Seul ViSP offre à la fois un haut degré de spécialisation et permet de considérer le problème de l'asservissement visuel de façon transversale allant de la commande au traitement d'image bas niveau et permettant de s'interfacer tant avec de véritables robots qu'avec des systèmes de simulation performants.

3 Les acteurs de l'ADT

3.1 Acteurs scientifiques

L'EPI Lagadic est l'acteur scientifique impliqué dans l'ADT ViSP. Ses membres produisent des briques logicielles innovantes de type "logiciel de recherche" à partir de la plate-forme logicielle ViSP. Certaines de ces briques ont ensuite été robustifiées et intégrées au logiciel (calcul de pose, estimation d'homographies, asservissement visuel 2D 1/2,...). D'autres, faute de ressources humaines sont laissées en friche sans pouvoir être pérennisées et intégrées (planification de trajectoire, utilisation des moments comme informations visuelles, ...).

Par ailleurs, ViSP est le point de départ de l'ensemble des développements de l'EPI Lagadic autour de ses actions contractuelles (projets pour l'ESA, pour la DGA, FP6 Pegase, etc).

Au niveau de l'Inria, la plate-forme ViSP est donc au cœur des développements réalisés dans l'EPI Lagadic et est utilisée par la totalité de ses membres (actuellement 16 personnes).

3.2 Acteurs techniques

L'acteur technique impliqué dans l'ADT ViSP est le SED de Rennes. Il intervient dans la mise à disposition d'un ingénieur permanent (Fabien Spindler) et par le support des services logiciels liés au développement. Fabien Spindler assure le support du logiciel, coordonne et participe aux développements de la plate-forme pour constamment l'améliorer. Depuis décembre 2006, il est épaulé par Anthony Saunier, ingénieur associé sur l'ODL ViSP qui va s'achever en décembre 2008.

3.3 Relation entre les acteurs scientifiques et techniques

L'ADT ViSP s'appuie sur Eric Marchand (CR1 Inria) responsable scientifique et Fabien Spindler (IR2 Inria) responsable technique dont les rôles sont précisés ci-dessous ;

Le responsable scientifique, Eric Marchand :

- est membre de l'EPI Lagadic depuis sa création. Ses travaux de recherche portent sur la vision robotique depuis ses travaux de doctorat. Il est l'auteur principal de la première version de ViSP ;
- participe et encadre des travaux de recherche exploitant les techniques d'asservissement visuel ;
- définit avec les autres scientifiques de l'EPI les briques logicielles matures pour être intégrées et pérennisées dans ViSP. Ces briques logicielles peuvent être issues de l'EPI ou faire partie de la littérature comme étant des algorithmes de l'état de l'art. Certaines de ces briques logicielles ont déjà été identifiées (voir le cahier des charges en 4.1). D'autres pourraient être fournies par des contributeurs extérieurs à l'EPI Lagadic ;
- arbitre les priorités des développements ;
- et participe à la promotion scientifique de ViSP.

Le responsable technique, Fabien Spindler :

- appartient au SED de Rennes et est détaché dans l'EPI Lagadic depuis sa création en janvier 2004. Il était auparavant responsable des plates-formes robotiques des équipes Temis puis Vista ;
- coordonne les développements et les améliorations apportées à la plate-forme ViSP ;
- participe aux développements logiciels ;
- veille à la qualité de la plate-forme (documentation, tests automatiques sur plusieurs cibles) ;
- veille au support de la plate-forme auprès de ses utilisateurs ;
- participe à la promotion technique de ViSP ;
- est le lien entre l'EPI et le SED ;
- encadre au quotidien l'ingénieur de l'ADT.

4 La description des activités de développement

4.1 Cahier des charges

Pour le moment, seules deux personnes sont fortement impliquées dans le développement du logiciel (Anthony Saunier (100%), ingénieur ODL jusqu'en décembre 2008 et Fabien Spindler (20 %), IR). Eric Marchand (CR) assure quant-à lui l'encadrement scientifique du projet à hauteur de 10 %. Ensemble, ils assurent le fonctionnement de ViSP sur les plates-formes de l'Inria Rennes, la maintenance et le support de ViSP dans un contexte hétérogène et multi plates-formes.

Pour mémoire, les objectifs visés dans cette ADT sont les suivants :

1. Renforcer la diffusion et la pérennisation de la plate-forme logicielle ViSP en tant que logiciel libre.
2. Réaliser des développements supplémentaires significatifs visant à intégrer les travaux de recherche récents de l'EPI Lagadic et à proposer à la communauté un outil plus performant comportant les dernières avancées dans ce domaine.

Pour remplir ces objectifs nous demandons des forces ingénieur estimées à 48 hommes par mois. Comme cela a déjà été dit, notre préférence va vers le recrutement d'un seul ingénieur sur une durée de quatre années. L'EPI et le SED de Rennes mettront également des forces permanentes (voir paragraphe 5.5). Ci-dessous, nous précisons les actions à mener pour atteindre nos objectifs.

4.1.1 Renforcer la diffusion et l'utilisation du logiciel

Le nombre actuel de téléchargements de la dernière version stable de ViSP (1005 depuis mai 2007) montre l'intérêt de la communauté pour ce logiciel et nous incite à poursuivre nos efforts dans la diffusion du logiciel. Malgré le nombre de téléchargements élevé, pour le moment nous avons assez peu de retours des utilisateurs. Même si certains laboratoires utilisent systématiquement ViSP, il nous est difficile de connaître le degré d'utilisation de la librairie. Plusieurs leviers sont à notre disposition pour améliorer la diffusion et l'utilisation de ViSP.

T1.1 : Amélioration de la documentation (3 HM). Le logiciel ViSP étant en constante évolution, il est nécessaire de maintenir une documentation à jour. Une nouvelle version du manuel utilisateur, plus complète et à jour par rapport à la première version diffusée en juillet 2005 [32], est en cours de rédaction. Elle a pour but d'aider l'utilisateur à installer et à utiliser le logiciel. Ce document devrait contribuer à renforcer l'utilisation du logiciel. Plus généralement, un effort permanent doit être mené pour mieux documenter le logiciel, tant au niveau de la mise à jour du manuel utilisateur que de la documentation du code. L'ingénieur ADT devra participer à l'enrichissement de ce manuel utilisateur et de la documentation html du code générée à l'aide de l'outil Doxygen, à l'amplification de la base d'exemples simples de type travaux pratiques, à la mise en place de tutoriels pour amener chaque nouvel utilisateur à découvrir simplement le logiciel. Pour que la documentation en ligne soit toujours à jour sur le site web de ViSP, celle-ci sera générée automatiquement toutes les nuits à partir de la base CVS et de l'outil Doxygen (voir Fig. 8). Plus généralement, l'amélioration de la documentation sera considérée comme une tâche de fond à mener durant toute la durée de l'ADT.

T1.2 : Renforcement des tests (3 HM). La qualité du logiciel diffusé est essentielle pour l'image de marque de l'EPI Lagadic et de l'institut. Actuellement, plusieurs tests sont déployés automatiquement sur les plates-formes Linux et Windows (voir Fig. 9, 10 et 11). Ils permettent d'observer la fonctionnalité et la non-régression du code. L'ingénieur devra déployer ces tests sur la plate-forme de test Pipol hébergée à Grenoble pour notamment assurer le fonctionnement du logiciel sur Mac OSX. Certaines parties de ViSP ne sont pas couvertes par des tests. Il s'agira dans un premier temps de les identifier, puis de développer de nouveaux tests unitaires. Les outils DART et CMake déjà exploités seront utilisés pour atteindre cet objectif et assurer la traçabilité du logiciel. Lorsque les tests seront mis en défaut, des actions correctrices devront immédiatement être mises en place pour garantir la qualité et la stabilité du code. Cette tâche sera également à réaliser comme tâche de fond.

T1.3 : Support et aide aux utilisateurs (3 HM). Le support aux utilisateurs sera assuré en continu tout au long de la période de l'ADT. Ceci inclut la maintenance du site web, la gestion des listes de diffusion et des forums de discussion, les réponses aux questions techniques, les corrections de bogues, etc. Cette tâche est à mener durant toute la période couverte par l'ADT.

T1.4 : Dynamisation de la structure d'échange autour de la forge (1 HM). Une structure d'échange hébergée sur la forge de l'Inria a été mise en place. Pour le moment, les forums de discussion (help et help.fr) sont assez peu actifs et le nombre de bogues relevés peu nombreux. Il s'agira de renforcer l'utilisation de ces outils pour dynamiser la diffusion du logiciel. Ce renforcement passe d'abord par la sensibilisation des utilisateurs de l'EPI à l'utilisation de ces outils. Le référencement de ViSP sur certains sites (comme celui d'Apple ou de CMake) permettrait également de mieux faire connaître le logiciel et de dynamiser sa diffusion.

T1.5 : Distribuer des binaires (1 HM). Pour le moment, ViSP est distribué exclusivement sous forme de code source. Pour bénéficier de l'ensemble des fonctionnalités de ViSP, il est d'abord nécessaire d'installer un certain nombre de librairies tierces (CMake, libdc1394, GTK, Coin, SoXt, ...). Ensuite, la configuration de ViSP (via CMake) et sa compilation doivent être réalisées. Ces étapes peuvent rebuter les utilisateurs potentiels. Diffuser également le cœur du logiciel sous forme de binaires à l'aide d'un "installeur" convivial permettrait à des non-spécialistes de faciliter l'installation et la découverte du logiciel. La plate-forme de portage pipol pourrait d'abord être exploitée pour compiler en statique le code de ViSP sur plusieurs cibles. Ensuite, l'outil de *packaging* CPack, venant avec l'outil CMake déjà utilisé pour permettre la configuration du logiciel dans un contexte multi plates-formes, pourrait-être exploité pour produire les *packages* debian, ubuntu, fedora, windows, ... pour ces cibles. Ces binaires n'incluront pas les librairies tierces utiles à l'exploitation de l'ensemble des fonctionnalités. Aussi, pour bénéficier de toutes les fonctionnalités de ViSP, les étapes de configuration et de compilation resteront à mener.

T1.6 : Capter la communauté des utilisateurs de Matlab (4 HM). La communauté en asservissement visuel comporte une grande partie d'automaticiens n'utilisant que Matlab. Pour capter ces utilisateurs potentiels, l'opportunité d'interfacer ViSP avec Matlab doit être étudiée. Dans un premier temps, une *toolbox* asservissement visuel pourrait être proposée aux utilisateurs de Matlab / Simulink. Elle exploiterait des fichiers MEX écrits en C, qui une fois

compilés, peuvent être utilisés dans Matlab de manière classique. Les fichiers MEX seraient alors des interfaces vers du code déjà présent dans ViSP. L'utilisation de MEX permettrait de s'appuyer sur la librairie ViSP existante sans avoir besoin de la reprogrammer dans le langage propre à Matlab. Cela permettrait en outre d'avoir une vitesse d'exécution plus rapide qu'en langage purement Matlab. L'interfaçage de ViSP au sein de Matlab apporterait indéniablement une plus value au logiciel. En fonction du succès ou non de cette *toolbox* asservissement visuel, d'autres fonctionnalités pourraient également être interfacées par la suite.

T1.7 : Capter la communauté des utilisateurs de Scilab (2 HM). La possibilité d'interfacer du code Scilab avec du code C existe également. En fonction du succès de la tâche précédente et en faisant un effort similaire, il sera possible de proposer une toolbox asservissement visuel aux utilisateurs de Scilab.

4.1.2 Réaliser des développements supplémentaires significatifs

Pour proposer à la communauté un outil toujours plus performant et l'inciter à utiliser le logiciel, de nouveaux développements doivent être menés. Ils visent également à mieux pérenniser les travaux de recherche de l'EPI Lagadic. L'ajout de nouvelles fonctionnalités est évidemment un point important pour mieux diffuser le logiciel.

T2.1 : Nouvelle classe image (2 HM). Actuellement, seules les images en niveaux de gris (où chaque pixel est codé sur un octet) ou en couleur RGB (ou chaque pixel est codé sur trois octets, respectivement pour les composantes rouge, vert et bleu) sont supportées dans ViSP. L'introduction d'autres formats d'images comme le HSV plus largement utilisé lorsqu'on souhaite traiter des images couleur n'est actuellement pas possible simplement. Pour mieux supporter la multitude de formats disponibles, il devient nécessaire de modifier la classe image existante pour la rendre plus modulaire. Par ailleurs, au niveau des entrées-sorties, seuls les formats non compressés PNM (PGM P5 pour les images en niveau de gris et PPM P6 pour les images couleur en RGB) sont supportés dans ViSP. Ces formats d'images frustres sont peu répandus dans les outils professionnels d'acquisition ou de visualisation de vidéos. Il est important d'étendre les entrées-sorties aux formats jpeg, tiff, png et mpeg, voire avi beaucoup plus largement utilisés. La réalisation de cette tâche passe par une analyse des classes images implémentées dans d'autres logiciels diffusés sous forme de logiciel libre, comme OpenCV ou la librairie CImg et par l'identification de librairies standard supportant ces format d'images ou de vidéos.

T2.2 : Exploitation de la couleur dans les images (2 HM). La plupart des caméras acquièrent des images couleur. Pour le moment, les algorithmes de traitement d'images présents dans ViSP utilisent des images en niveaux de gris. Il serait intéressant de faire évoluer certains de ces algorithmes pour exploiter la couleur, riche en informations. Il s'agit notamment d'algorithmes de suivi d'objets basés sur une binarisation de l'image et déjà présents dans ViSP.

T2.3 : Optimisation du code (3 HM). Le logiciel ViSP est dédié aux applications temps réel de vision robotique cadencées à la fréquence d'acquisition des images, typiquement 50 Hz. Il apparaît donc crucial de disposer d'algorithmes de vision et de traitement d'images optimisés et fiables. Des premiers tests ont montré que certaines parties du logiciel mériteraient d'être optimisées davantage. Des outils de profiling seront utilisés pour identifier les parties critiques et

les optimiser. Avec la génération des processeurs multi-cœurs, il devient intéressant d'exploiter plus largement le *multi-threading* et les options des compilateurs dédiées à ces architectures. Les capacités des cartes graphiques actuelles pourront également être explorées pour voir s'il n'est pas possible d'améliorer à moindre coût les performances du logiciel en utilisant par exemple les accélérations matérielles ou les processeurs graphiques (GPU).

T2.4 : Améliorer les capacités de simulation (3 HM). L'utilisation de ViSP en tant que simulateur pour l'asservissement visuel nous paraît fondamental pour assurer une vaste diffusion au logiciel. Le simulateur actuel fonctionne sur des plates-formes hétérogènes, mais ne nous donne pas entière satisfaction en terme de performances et de fonctionnalités.

Pour pouvoir utiliser ce simulateur, il est nécessaire d'installer au préalable la librairie Coin (Open Inventor) et un *front end* SoXt, ou SoWin ou SoQt. L'utilisation de ces librairies tierces empêche de distribuer les binaires de ViSP en incluant cette fonctionnalité de simulation. L'intégration d'un simulateur rudimentaire type fil de fer ne nécessitant aucune librairie tierce permettrait de lever ce verrou.

Par ailleurs, la caméra virtuelle utilisée pour la simulation peut pour le moment évoluer dans un monde virtuel infini. Il serait utile de pouvoir l'associer à un robot ayant des caractéristiques physiques (butées, vitesses maximales, articulations, modèles géométriques) pour obtenir des résultats plus réalistes.

T2.5 : Interfaçage avec le robot médical Echo3D de l'Inria Rennes (1 HM). Un robot d'acquisition d'échographies 3D (Echo3D), commun aux EPI Lagadic et Visages permet de valider les travaux de recherche menés en robotique médicale. Ce robot à six degrés de liberté porte un capteur d'effort et une sonde échographique 2D. Il s'agira d'interfacer ce matériel à ViSP pour simplifier son utilisation.

T2.6 : Interfaçage de nouvelles caméras (1 HM). De plus en plus d'utilisateurs disposent de caméras USB bas-coût. Ces caméras sont actuellement supportées sous Windows via DirectShow. Il serait utile de les interfacer sous Linux en utilisant par exemple la librairie Video For Linux (V4L). Par ailleurs, la librairie DV est également à interfacer pour permettre l'acquisition de vidéos à l'aide de caméscopes.

T2.7 : Intégration d'algorithmes classiques de vision par ordinateur (5 HM). L'intégration d'algorithmes "classiques" de vision par ordinateur décrits dans la littérature apporterait indéniablement une forte valeur ajoutée. Il pourrait s'agir d'algorithmes de suivi dont nous ne disposons pas pour le moment (par exemple les Sift), ou d'algorithmes déjà développés dans l'EPI mais non intégrés dans ViSP (par exemple le suivi de contours actifs). En interne, cela permettrait de mutualiser les développements et en externe d'augmenter la visibilité du logiciel.

Plus généralement, l'ajout de nouvelles fonctionnalités via l'intégration des méthodes développées dans d'autres laboratoires, ou via des développements propres est envisagé pour compléter les capacités de la librairie ViSP.

Parmi les développements possibles, nous pouvons citer l'ajout d'algorithmes de reconnaissance d'objets pour automatiser l'initialisation du suivi, de nouvelles lois de commande, l'interfaçage avec certaines fonctionnalités de la librairie OpenCV d'Intel, etc.

T2.8 : Intégration de méthodes de planification de trajectoires dans l'image (2 HM). Les lois de commande actuellement intégrées dans ViSP utilisent une consigne fixe (par exemple, un objet doit apparaître à tel endroit dans l'image). Quand l'erreur de positionnement ou de suivi initiale est forte, il est pourtant plus judicieux de générer des trajectoires de consigne à suivre afin de rendre les lois de commande plus robustes aux erreurs de modélisation et de calibration. Nous souhaitons donc intégrer dans ViSP les méthodes de planification et de suivi de trajectoires qui ont été devéloppées dans [37].

T2.9 : Prise en compte des capteurs de vision omnidirectionnelle (3 HM). Les seuls capteurs de vision considérés actuellement dans ViSP sont des caméras classiques régies par un modèle de projection perspective. Dans de nombreuses applications, en robotique mobile et aérienne notamment, on trouve à présent de plus en plus de nouveaux capteurs à très grand champ de vue, tels les caméras fish-eye ou les capteurs de vision omnidirectionnelle couplant une caméra classique à des miroirs de diverses formes (conique, parabolique, ...). Ces capteurs ne sont pas régis par un modèle de projection perspective, mais par des modèles plus complexes. Nous prévoyons donc d'étendre ViSP à toute la famille des capteurs régis par un modèle approximatif de projection à centre unique, ce qui inclut les caméras classiques, les caméras fish-eye et les capteurs de vision omnidirectionnelle les plus courants.

T2.10 : Valorisation de la plate-forme par la réalisation de démonstrations (5 HM). La réalisation de l'ensemble des tâches précédentes permettra de renforcer très nettement les performances de ViSP. La valorisation de ces travaux passe également en interne par la réalisation de nouvelles démonstrations sur les plates-formes de vision robotique de l'Inria-Rennes. Elles permettront de présenter à nos visiteurs des travaux de recherche innovants et de renforcer la visibilité et la notoriété de l'EPI. Auprès d'un public de scientifiques, ces démonstrations participent très efficacement à la diffusion de nos résultats de recherche et à la valorisation de notre savoir faire autour de ViSP. Les industriels constatent eux par une démonstration que nos travaux ont été validés sur des applications réelles et sur du matériel industriel. Ils sont alors plus enclins à s'impliquer dans des collaborations de transfert technologique. Les démonstrations servent évidemment comme outil de communication pour un public moins averti.

T2.11 : Intégration des moments comme informations visuelles (2 HM). De nombreuses informations visuelles peuvent être utilisées en entrée de lois de commande d'asservissement visuel. ViSP en contient déjà un bon nombre (coordonnées de points, paramètres représentant des droites et des ellipses, entités 3D, ...). En raison de leur très bonnes propriétés [43], nous souhaitons y inclure les moments que l'on peut calculer à partir d'un nuage de points ou de régions segmentées dans l'image. Cela nécessite d'intégrer les algorithmes pour calculer la valeur des divers moments possibles (moments centrés, normalisés, invariants de Hu, invariants mis en évidence dans [43], combinaison adéquate de moments), mais aussi les matrices d'interaction associées.

4.2 Méthodes et outils de développement

La mise en œuvre des objectifs décrits dans cette ADT nécessite d'implémenter du logiciel, de le déverminer et de le tester de manière continue sur des cibles hétérogènes, à la fois en terme de système d'exploitation (Linux, Windows, MacOSX), mais également en terme de compilateur (g++ , Visual Studio .NET, Visual Studio 2005, ...). Par ailleurs, la réalisation

des objectifs de l'ADT va permettre la constitution progressive d'une communauté de développeurs. Les développements seront alors potentiellement réalisés par plusieurs personnes réparties sur plusieurs sites.

Pour prendre en compte ces éléments, notre processus de développement, résumé figure 8 et déjà en grande partie opérationnel depuis juin 2006, s'appuie sur des logiciels open-source qui permettent de faciliter le développement multi plate-forme, la génération automatique de documentation, la mise en place de tests automatiques en continu et la présentation des résultats de ces traitements sur un site web public.

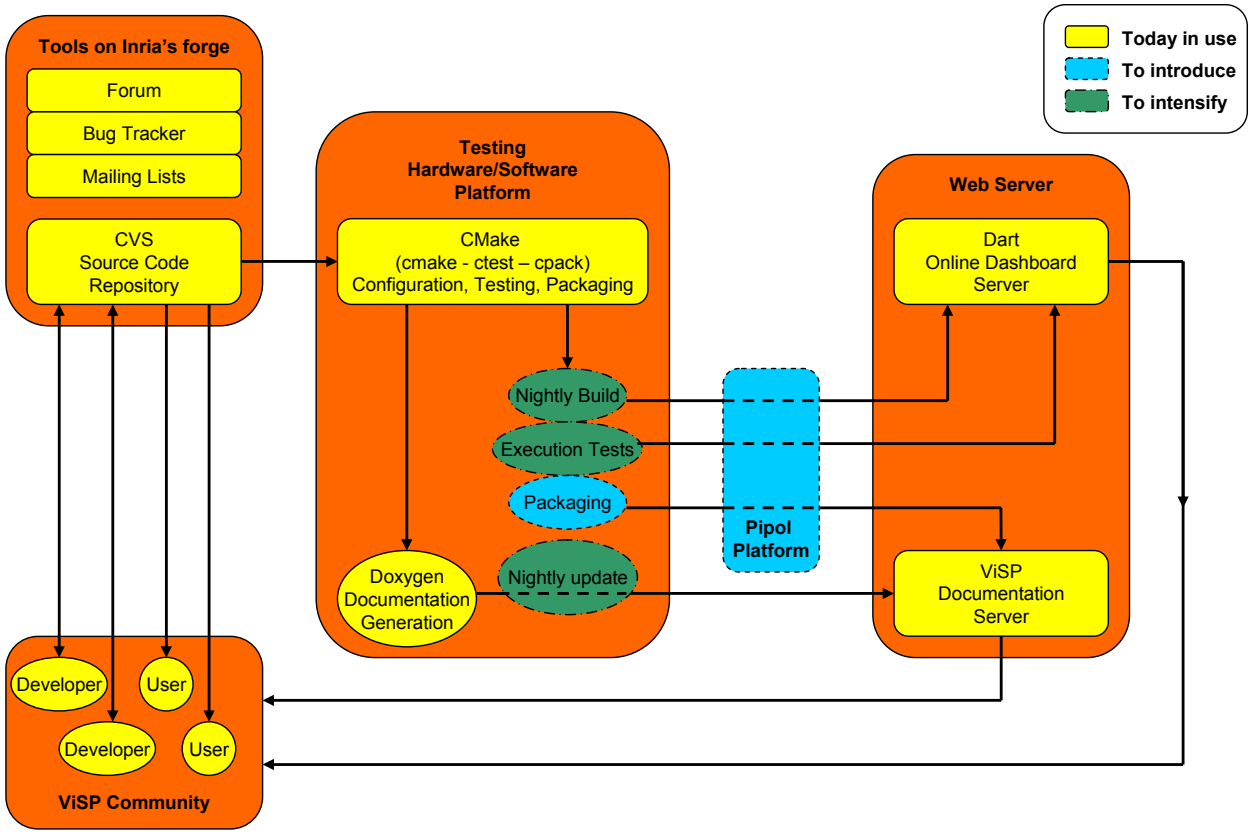
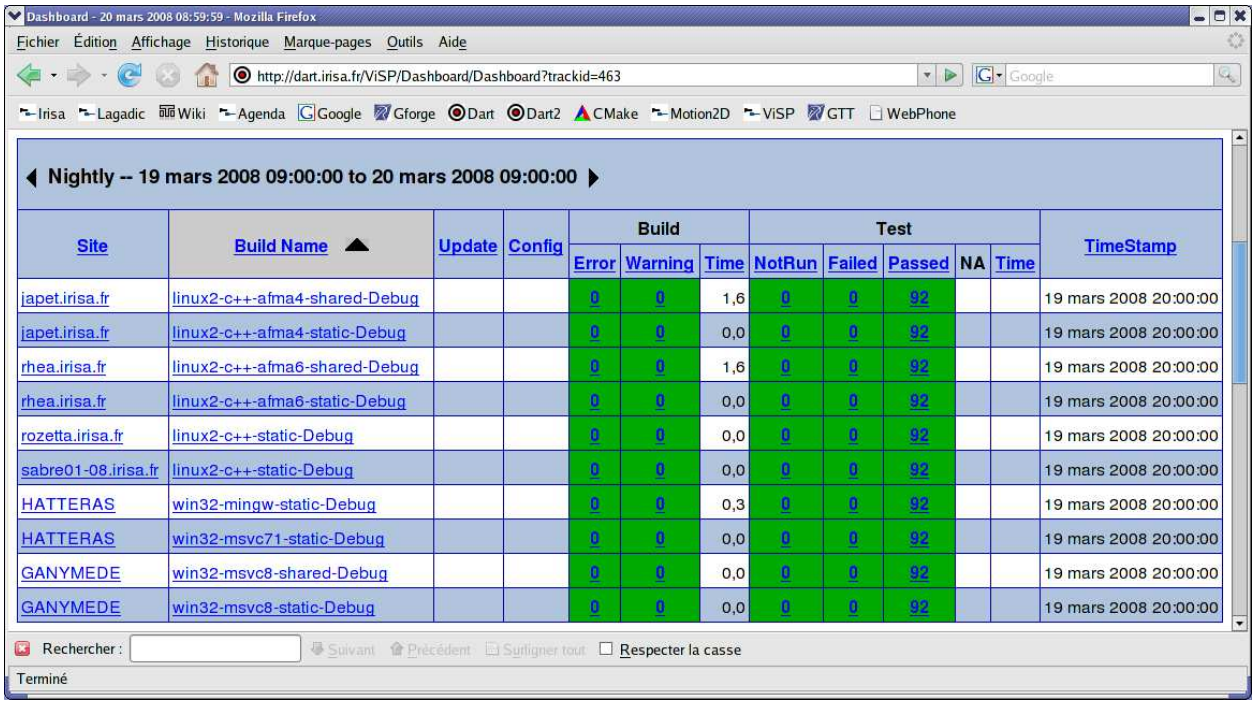


FIG. 8 – Outils mis en œuvre durant le processus de développement pour assurer la qualité du logiciel produit. Cette figure présente les outils déjà utilisés depuis juin 2006, ceux qui seront introduits durant l'ADT (comme `cpack`) et ceux dont l'utilisation sera intensifiée pour notamment permettre la mise à jour automatique de la documentation sur le serveur web de ViSP et déployer davantage de tests.

La figure 8 illustre les interactions entre les outils supportant notre processus de développement. Depuis que ViSP est un projet public sur la forge de l'Inria, plusieurs développeurs apportent leurs contributions dans la base CVS (CVS *repository*). CVS garde trace des modifications apportées au code (qui, quoi, pourquoi et quand) et permet de produire certains indicateurs comme ceux présentés Fig. 5 et 6. Le processus de *build* est controlé par l'outil multi plate-forme CMake. CMake met à disposition de l'utilisateur trois binaires (`cmake`,

`ctest` et `cpack`) tous exploités dans l'ADT. Ils sont utilisés pour assurer la portabilité du logiciel sur des plates-formes hétérogènes. `cmake` est unique dans le sens où il ne remplace pas les fichiers de configuration comme les Makefile ou les *Workspace* Windows. Au contraire, `cmake` va générer ces fichiers de configuration à partir de fichiers CMakeLists.txt indépendants de la plate-forme cible. `cmake` va ensuite exploiter les outils classiques de compilation (make et gcc, Visual Studio, ...) pour réaliser les étapes de compilation et d'édition de liens. Dart quant-à lui est là pour coordonner le processus de test. Associé au binaire `ctest` (intégré dans CMake), il va recenser les erreurs de compilation et d'édition de liens (voir Fig. 9), les problèmes de gestion de la mémoire en exploitant des outils comme Valgrind ou Purify (voir Fig. 10), lancer les tests de couverture (voir Fig. 11) et les tests d'exécution du logiciel (quatre vingt douze actuellement, voir Fig. 9).



Site	Build Name	Update	Config	Build			Test					TimeStamp
				Error	Warning	Time	NotRun	Failed	Passed	NA	Time	
japet.irisa.fr	linux2-c++-afma4-shared-Debug			0	0	1,6	0	0	92			19 mars 2008 20:00:00
japet.irisa.fr	linux2-c++-afma4-static-Debug			0	0	0,0	0	0	92			19 mars 2008 20:00:00
rhea.irisa.fr	linux2-c++-afma6-shared-Debug			0	0	1,6	0	0	92			19 mars 2008 20:00:00
rhea.irisa.fr	linux2-c++-afma6-static-Debug			0	0	0,0	0	0	92			19 mars 2008 20:00:00
rozetta.irisa.fr	linux2-c++-static-Debug			0	0	0,0	0	0	92			19 mars 2008 20:00:00
sabre01-08.irisa.fr	linux2-c++-static-Debug			0	0	0,0	0	0	92			19 mars 2008 20:00:00
HATTERAS	win32-mingw-static-Debug			0	0	0,3	0	0	92			19 mars 2008 20:00:00
HATTERAS	win32-msvc71-static-Debug			0	0	0,0	0	0	92			19 mars 2008 20:00:00
GANYMEDE	win32-msvc8-shared-Debug			0	0	0,0	0	0	92			19 mars 2008 20:00:00
GANYMEDE	win32-msvc8-static-Debug			0	0	0,0	0	0	92			19 mars 2008 20:00:00

FIG. 9 – Exemple de *dashboard* résumant le résultat des tests de compilation et d'exécution obtenus sur plusieurs postes de travail (Linux et Windows) dans différentes configuration (avec ou sans robot, avec production de librairie statique ou dynamique, en mode debug ou *release*).

Le client Dart poste ensuite les résultats de ces tests sur le serveur Dart via un protocole XML sous forme de pages web. Ces pages correspondent à un *dashboard* résumant l'état du logiciel. Ces tests sont lancés automatiquement toutes les nuits à partir du code présent dans la base CVS. L'outil Doxygen est utilisé pour générer la documentation html du logiciel à partir du code source et de sa documentation embarquée. CMake sera utilisé pour appeler Doxygen automatiquement toutes les nuits pour mettre à jour la documentation du code sur le web et ainsi proposer aux utilisateurs et aux développeurs une documentation en permanence à jour.

`cpack` sera mis en œuvre durant l'ADT pour répondre aux besoins de la tâche T1.5 visant à distribuer des binaires (voir paragraphe 4.1.1). `cpack` permet de créer des "installeurs" conviviaux de binaires sur les plates-formes Linux (*package* debian, ubuntu, fedora,...) et Windows (à partir de l'outil NSIS).

La forge de l'Inria continuera à être utilisée pour exploiter hormis CVS, son outil de suivi

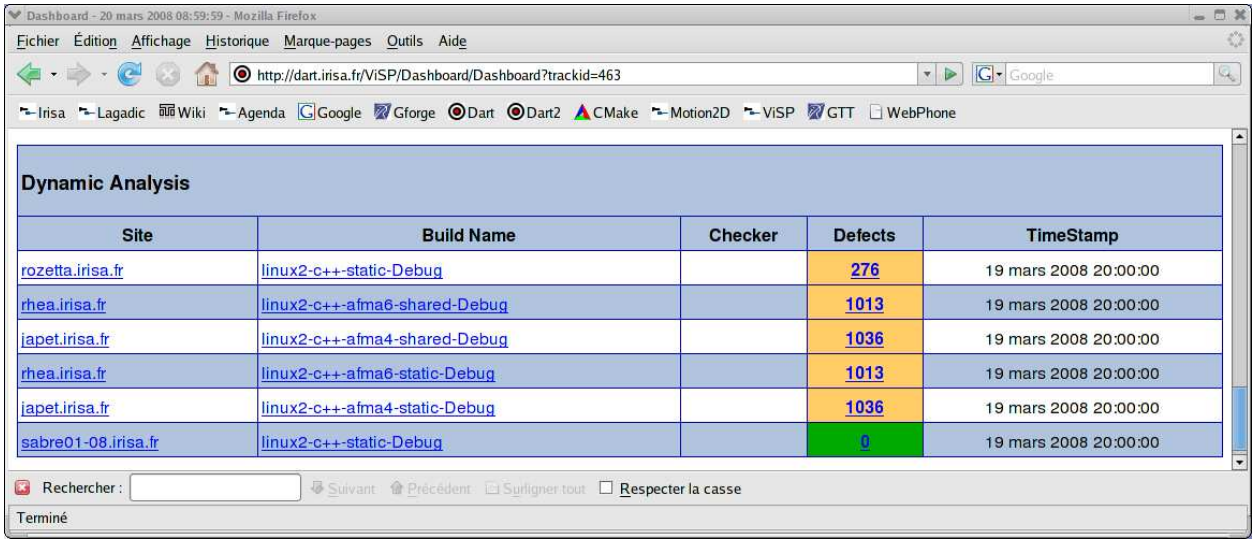


FIG. 10 – Exemple de *dashboard* résumant les tests liés à l'analyse de la mémoire exploitée par le logiciel à l'aide de l'outil Valgrind.

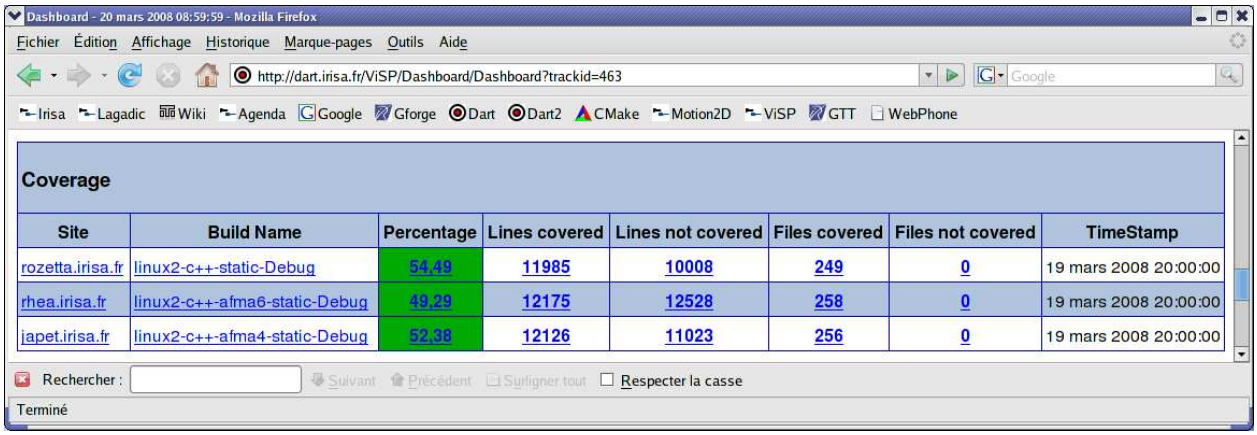


FIG. 11 – Exemple de *dashboard* résumant les tests de couverture de ViSP.

de bogues, ses *mailings-lists*, ses forums de discussion et ses outils de statistiques utiles à la fourniture d'indicateurs (voir paragraphe 2.2.2).

Mis à part la génération automatique toutes les nuits de la documentation à mettre en œuvre dans la tache T1.1 (voir paragraphe 4.1) et l'utilisation de `cpack` pour générer les "installeurs" de binaires correspondant à une version de ViSP ne faisant pas appel à des librairies tierces, le processus de développement décrit précédemment est déjà opérationnel depuis juin 2006.

Dès qu'elle sera opérationnelle sur les environnements Windows et Mac OSX, la plate-forme de portage Pipol hébergée à Grenoble sera utilisée pour déployer des tests automatiques sur de nouvelles cibles et ainsi veiller à la qualité du logiciel dans un contexte multi plates-formes. Pipol sera également exploitée de manière plus soutenue durant les campagnes de tests intensifs précédent chaque *release*.

Ce processus de développement est essentiel. Il permet en permanence de vérifier le bon fonctionnement du logiciel, d'apporter des corrections, de valider le développement, d'assurer sa traçabilité et de fournir des indicateurs de qualité.

4.3 Échéancier

Le tableau (Tab. 1) récapitule les tâches décrites au paragraphe 4.1 et les ressources en hommes par mois nécessaires pour les réaliser. Même si des durées sont mentionnées, certaines de ces tâches sont à réaliser de manière épisodique et en parallèle d'autres tâches durant toute la durée de l'ADT. Il s'agit notamment des tâches permettant de renforcer la diffusion du logiciel ; T1.1, T1.2 et T1.3. D'autres, plus liées à l'ajout de nouveaux développements seront à reprendre selon un cycle itératif (implémentation, tests, validation sur système réel, corrections...). L'ordonnancement de ces tâches est précisé Fig. 12.

4.4 Planning et jalons

Le diagramme de Gant présenté Fig. 12 précise l'organisation des développements et l'ordonnancement en fonction du temps des tâches décrites au paragraphe 4.3. Ce diagramme couvre une période de quatre ans pour préciser les 48 hommes par mois demandés. Six *releases* seront proposées aux utilisateurs. Les deux premières années nous prévoyons la livraison d'une *release* par an. Ensuite, les *releases* seront proposées au rythme de deux par an. S'il n'était pas possible de recruter un seul ingénieur durant quatre ans, en faisant l'hypothèse de deux ingénieurs durant deux ans, les tâches plannifiées durant les années 1 et 2 seront à mener en parallèle de celles planifiées les années 3 et 4. Dans ce cas, seules quatre *releases* seront réalisées ; une tous les six mois.

Mis à part les tests automatiques mis en œuvre chaque jour (voir paragraphe 4.2), avant chaque *release*, une campagne de tests approfondis sera menée. Des *beta* testeurs, internes à l'EPI, mais également externes (voir paragraphe 7.2), seront sollicités.

A chaque *release* correspondra un guide utilisateur mis à jour pour préciser et détailler les nouvelles fonctionnalités ou modifications apportées à ViSP.

Les *releases* intégrant des développements supplémentaires significatifs feront l'objet d'une mise à jour des dépôts APP réalisés précédemment.

4.5 Risques techniques

Vues nos compétences dans le domaine de l'asservissement visuel, le bon déroulement de l'ODL actuelle et le fait que la version courante de ViSP est fiable et stable, nous sommes très

Tâche	Ressources
Formation et apprentissage	2 HM
Renforcer la diffusion et l'utilisation du logiciel	
T1.1 : Amélioration de la documentation	3 HM
T1.2 : Renforcement des tests	3 HM
T1.3 : Support et aide aux utilisateurs	3 HM
T1.4 : Dynamisation de la structure d'échange autour de la forge	1 HM
T1.5 : Distribuer des binaires	1 HM
T1.6 : Capter la communauté des utilisateurs de Matlab	4 HM
T1.7 : Capter la communauté des utilisateurs de Scilab	2 HM
Réaliser des développements supplémentaires significatifs	
T2.1 : Nouvelle classe image	2 HM
T2.2 : Exploitation de la couleur dans les images	2 HM
T2.3 : Optimisation du code	3 HM
T2.4 : Améliorer les capacités de simulation	3 HM
T2.5 : Interfaçage avec le robot médical Echo3D	1 HM
T2.6 : Interfaçage de nouvelles caméras	1 HM
T2.7 : Intégration d'algorithmes classiques de vision par ordinateur	5 HM
T2.8 : Intégration de méthodes de planification de trajectoires dans l'image	2 HM
T2.9 : Prise en compte des capteurs de vision omnidirectionnelle	3 HM
T2.10 : Valorisation de la plate-forme par la réalisation de démonstrations	5 HM
T2.11 : Intégration des moments comme informations visuelles	2 HM
Total	48 HM

TAB. 1 – Tâches à effectuer pour atteindre les objectifs de l'ADT. Pour chaque tâche les ressources en hommes par mois sont affichées.

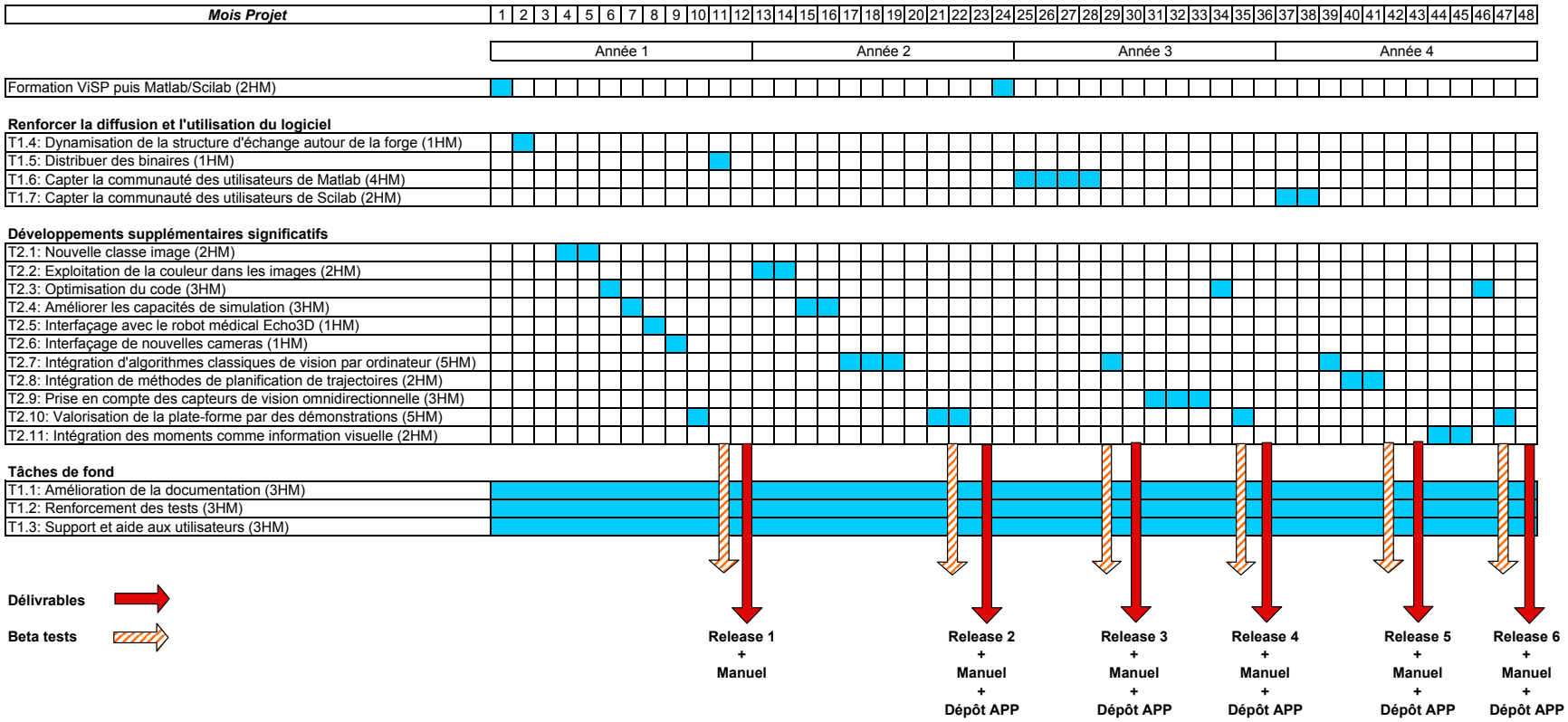


FIG. 12 – Planning prévisionnel des tâches nécessaires à la réalisation des objectifs de l'ADT. Six *releases* seront livrées. Chaque *release* sera précédée d'une campagne de tests approfondis réalisés par un groupe de *beta* testeurs. Chaque *release* sera accompagnée d'un guide utilisateur mis à jour.

confiants en ce qui concerne le succès de cette ADT. Nous n'avons pas été capables d'identifier de risque majeur, si ce n'est bien entendu, un peu probable échec sur la qualité du recrutement à effectuer.

Les autres risques mineurs possibles portent sur :

– une sous estimation de la durée nécessaire pour réaliser certaines tâches. Dans ce cas, des tâches moins prioritaires (telles par exemple T2.11 décrite au paragraphe 4.1.2) seraient abandonnées.
– une difficulté à interfacer ViSP avec Matlab et Scilab. Le risque porte davantage sur le compromis à trouver entre la modularité de ViSP et la définition d'un nombre limité de fonctions Matlab appelant ViSP, que sur des aspects techniques proprement dits.

5 Les moyens de l'ADT

La mise en œuvre de l'ADT repose sur des moyens humains (scientifiques et techniques) et matériels (équipement informatique) détaillés dans les paragraphes suivants.

5.1 Moyens humains pour les activités scientifiques

Les moyens scientifiques en interaction avec l'ADT relèvent de l'EPI Lagadic. Les personnels permanents impliqués sont :

– ÉRIC MARCHAND, Chargé de Recherche Inria, auteur principal des versions préliminaires du logiciel ViSP, responsable scientifique de l'ADT ;
– FRANÇOIS CHAUMETTE, Directeur de Recherche Inria, responsable de l'EPI Lagadic, auteur de certains modules du logiciel, apportera son expertise dans le domaine de l'asservissement visuel.
– ALEXANDRE KRUPA, Chargé de Recherche Inria menant une activité de recherche dans le domaine de la robotique médicale participera à la définition des besoins à mettre en œuvre dans la réalisation de la tâche T2.5 décrite dans le paragraphe 4.1.2 et à sa validation ;

Les doctorants et post-doctorants de l'EPI seront les premiers utilisateurs de ViSP. Certains d'entre eux seront des *beta* testeurs du logiciel. D'autres souhaiteront probablement contribuer occasionnellement pour adapter le logiciel à leur besoin spécifique. D'autres encore solliciteront l'équipe technique pour des questions de support. La proximité de ces utilisateurs est un avantage dont il faudra tirer parti pour anticiper les questions pouvant provenir des utilisateurs externes à l'EPI.

5.2 Moyens humains pour les activités techniques

Fabien Spindler, Ingénieur de Recherche Inria, membre du SED de Rennes, responsable des plates-formes matérielles de vision robotique de l'Inria Rennes, responsable technique de la plate-forme logicielle ViSP et auteur de certains modules du logiciel sera le responsable technique de l'ADT. C'est lui qui coordonnera l'ensemble des développements menés dans l'ADT. Il aura sous sa responsabilité une équipe technique composée d'Anthony Saunier, Ingénieur Associé sur l'ODL ViSP jusqu'au 15 décembre 2008 et les forces en ingénieur à recruter à partir de septembre 2008 correspondant aux 48 hommes par mois nécessaires pour répondre aux objectifs de l'ADT.

Pour réaliser cette ADT, nous rechercherons un ingénieur informaticien ayant des compétences en traitement d'images et, si possible, en robotique et vision par ordinateur. Une *excellente* connaissance du C++ et de la programmation sous Unix et Windows sera impérative. La connaissance de Matlab ou de Scilab est également requise. La personne recrutée devra en outre avoir une très bonne maîtrise de la langue anglaise (écrit et oral). Elle devra également faire preuve d'autonomie et de capacité à travailler en équipe.

5.3 Moyens matériels

Les moyens matériels nécessaires à l'ADT correspondent à deux postes de travail pour l'ingénieur à recruter : un PC fixe sous Linux et un PC fixe sous Windows. Tous deux seront équipés d'une carte graphique NVidia GPUs compatible avec cuda. Ces matériels seront exploités pour mener à bien les développements spécifiques sur ces deux environnements et mettre en œuvre les tests associés. Une première évaluation du coût de ce matériel est de 8.000 €.

5.4 Budget de fonctionnement

Nous proposons que l'évaluation du bon fonctionnement de l'ADT soit en partie menée par des spécialistes du domaine extérieurs à l'EPI Lagadic (voir 7.2). Leurs frais de mission seront évidemment à prendre en charge. Le coût annuel de ces missions est estimé à 1.500 € par personne, soit potentiellement 3.000 € ou 4.500 € en fonction du nombre d'évaluateurs.

L'interfaçage de ViSP avec Matlab et Scilab demande des compétences pointues dans l'utilisation de ces deux outils. Une formation autour de Matlab-Simulink serait utile pour démarrer sur de bonnes bases. Son coût est estimé à 4.000 € pour deux personnes. Il en va de même avec Scilab avec un coût de formation estimé à 3.000 €.

5.5 Récapitulatifs de moyens de l'ADT

Le tableau 2 récapitule les moyens demandés pour réaliser les objectifs de l'ADT.

Ressources en ingénieur	48 HM
Equipement	8.000 €
Fonctionnement	11.500 €
Total	19.500 €

TAB. 2 – Moyens demandés pour l'ADT.

Le tableau 3 présente les moyens mis en face par l'EPI Lagadic et le SED de Rennes pour coordonner, encadrer et participer aux développements de ViSP.

6 Autres points

6.1 Politique de protection intellectuelle et industrielle

Depuis 1999, les versions majeures de ViSP ont fait l'objet de dépôts à l'Agence de Protection des Programmes (APP). Depuis juin 2006, ViSP est hébergé sur la forge de l'Inria et

Ressources en ingénieur fournies par le SED	10 HM
Ressources en chercheurs (CR) fournies par l'EPI	5 HM
Ressources en chercheur (DR) fournies par l'EPI	2,5 HM
Total	17,5 HM

TAB. 3 – Ressources mises à disposition par l'EPI et le SED de l'Inria Rennes ; Fabien Spindler (IR), responsable technique, sera impliqué à hauteur de 20 %, Eric Marchand (CR), responsable scientifique sera impliqué à hauteur de 10 %, et François Chaumette (DR) à hauteur de 5 %.

diffusé en tant que logiciel libre sous licence QPL. Il est prévu de mettre à jour le dépôt avec le contenu de la prochaine version stable de ViSP (ViSP-2.4.2) prévue début avril 2008. Il va de soi que les versions de ViSP intégrant des fonctionnalités supplémentaires significatives et développées dans le cadre de l'ADT seront également déposées à l'APP.

6.2 Politique de diffusion

Deux forums de discussion, l'un en français, l'autre en anglais, un outil de suivi de bogues, une *mailing-list* et le gestionnaire de version CVS ont été déployés pour améliorer la diffusion du logiciel. Le site web dédié au logiciel a été sensiblement enrichi. Le travail de fond mené pour améliorer la qualité de la plate-forme commence à porter ses fruits puisque plus de 1005 téléchargements de la dernière version ViSP-2.4.1 datant de mai 2007 ont été enregistrés (Fig. 5), sans compter ceux réalisés directement depuis la base CVS (Fig. 6). A titre d'information, ViSP-2.2.0 datée de juin 2006 et ViSP-2.4.0 datée de février 2007 ont été téléchargés respectivement 251 et 473 fois.

Aujourd'hui, un certain nombre de laboratoires travaillant dans le domaine de la vision robotique utilisent ViSP. Comme son téléchargement est libre, il nous est impossible de savoir qui charge et qui en fait quoi. Depuis sa première diffusion en juin 2006, nous avons tout de même pu identifier (suite à des mél adressés aux auteurs, des demandes d'aide postées sur les forums, des discussions) : en France, le Lasmea et l'IFMA à Clermont-Ferrand, le CEA à Fontenay aux Roses ; aux Etats-Unis, l'Université Johns Hopkins et tout récemment Google ; au Japon, le JRL (Joint Robotic Laboratory) ; en Italie, l'Université de Sienne ; en Espagne, la société Daem Interactive, et l'Université de Jaume ; en Chine, l'Université de Shanghai, le Harbin Institute of Technology, et le Bejing Institute of Technology ; au Portugal, l'Université de Lisbonne, l'IST (Instituto Superior Técnico) ; en Inde, l'India Institute of Information Technology ; au Liban, le Centre Universitaire de Technologie ; en Hongrie, le Budapest University of Technology and Economics ; et enfin en Corée, l'Intelligent System Research Center in Suwon.

Ces efforts doivent être poursuivis et amplifiés pour fidéliser les utilisateurs et assurer le support, ajouter de nouvelles fonctionnalités et renforcer la diffusion du logiciel.

6.3 Politique de transfert

ViSP est utilisé par l'ensemble des membres de l'EPI Lagadic (16 personnes) pour mener à bien ses activités de recherche. Dans le cadre de ses actions contractuelles, depuis que ViSP est multi plates-formes, ViSP sert systématiquement de librairie de base dans les développements réalisés. En 2007, il a ainsi été utilisé lors d'actions de valorisation avec France Telecom R&D dans le cadre d'un contrat d'étude et de recherche, avec Dassault Aviation dans le cadre du

projet européen FP6 Pegase, avec Thalès Optronics dans le cadre du PEA Tarot de la DGA et avec l'agence spatiale européenne (ESA) dans le cadre de l'ITT Vimanco. L'utilisation de ViSP nous permet de factoriser et de mutualiser nos développements. Ceci est notamment possible depuis que ViSP a été porté sous Windows, un OS largement plébiscité par les industriels.

A l'avenir, ViSP sera toujours au cœur des développements réalisés dans l'EPI et continuera à être transféré au sein d'actions contractuelles. A terme, l'EPI n'exclut pas la conclusion d'une licence commerciale avec l'un de ses partenaires.

7 Evaluation

7.1 Critères de succès

En interne, il est indéniable que la librairie ViSP est déjà un succès. Comme cela a déjà été dit, l'ensemble des membres de l'EPI Lagadic utilise ce logiciel pour mener à bien les travaux de recherche ou les actions contractuelles de transfert. Cette librairie permet de mutualiser certains de nos développements et donc d'être plus efficaces. Il reste malgré tout de nombreux travaux de recherche innovants menés au sein de l'EPI Lagadic ayant abouti à des prototypes de "type recherche" de briques logicielles non pérennisées dans ViSP. Ces fonctionnalités gagneraient à être validées plus largement et intégrées dans la version diffusée de ViSP. D'autres faisant partie de l'état de l'art seraient également très utiles.

En externe, le succès de ViSP passe par un accroissement de la diffusion du logiciel. Des laboratoires clairement identifiés (voir paragraphe 6.2) utilisent déjà ViSP de manière intensive, d'autres de manière plus occasionnelle.

De manière plus pragmatique, le succès de l'ADT peut-être évalué à partir de critères qui :
- reflètent de l'activité du projet,
- rendent compte de la portée du logiciel et de son utilisation en dehors de l'EPI Lagadic,
- et plus généralement indiquent si les objectifs visés et les tâches identifiées en section 4 ont été atteints.

Ainsi, l'activité du projet peut-être mesurée en mettant en relation les dates clés du développement (voir Fig. 2) avec :
- les ressources en hommes par mois attribuées au projet (voir Fig. 3),
- la mise en œuvre des nouvelles fonctionnalités affichées dans l'ADT (voir pararagraphe 4.1),
- l'évolution du nombre de lignes de code pour mesurer le dynamisme des développements (voir Fig. 4),
- le nombre de bogues recensés sur la forge (actuellement 65 bogues auxquels des correctifs ont été apportés sont répertoriés),
- et le nombre de tests pour donner une idée de sa qualité et de ses performances qui doivent aller en s'améliorant (actuellement 92 tests existent).

L'impact de ViSP dans la communauté scientifique peut-être évalué à partir :
- du nombre de publications référençant ViSP (voir Tab. 4),
- du nombre de publication référençant ViSP et mentionnant explicitement que les développements ont été réalisés à l'aide de ViSP (voir Tab. 4),

– le taux d'activité dans les forums de discussions pour percevoir l'intérêt du logiciel pour les utilisateurs (actuellement 9 *threads* sont répertoriés),
– les contributions des utilisateurs extérieurs à l'EPI (1 patch a été proposé en 2006),
– le nombre de téléchargements du logiciel (voir Fig. 5 et Fig. 6) et des documentations associées (voir Fig. 7).

Publications		2006	2007	2008
Articles de journaux :	IJRR ISR TVCG	 ([6])	[2] ([41])	 [38]
Articles de conférence :	ICRA IROS	 [13] [14]	[12] [21] ([25] [28] [44]) [1] [35] [39]	([36])
Workshops		[5] [9] [19]		
Chapitres de livre			[17]	
Rapports de stage			[23]	

TAB. 4 – Liste des publications connues référençant ViSP au travers de la publication [33]. Celles entre parenthèses sont des publications de l'EPI Lagadic. Ces publications ont été identifiées soit à partir de Google, soit à partir des actes des conférences. Mis à part [5], [17] et [19] toutes ces publications mentionnent l'utilisation de ViSP.

Certains de ces indicateurs seront difficiles à mettre en œuvre et d'autres donneront une idée partielle de l'influence de ViSP. Ainsi, pour des questions de confidentialité, il sera notamment difficile de mesurer le taux d'utilisation de ViSP chez les industriels avant qu'ils souhaitent faire une utilisation commerciale, moment à partir duquel ils sont censés nous contacter pour acquérir une licence.

7.2 Evaluateurs extérieurs

Nous proposons de faire appel à des évaluateurs externes à l'EPI Lagadic, tels Philippe Martinet, responsable du Groupe de Recherche en Automatique, VIsion et Robotique (GRAVIR) au Lasmea à Clermont-Ferrand, Jacques Gangloff, professeur à l'Université Louis Pasteur de Strasbourg et membre de l'Equipe Automatique Vision et Robotique (EAVR) du LSIIT, et enfin, Patrick Rives, responsable de l'EPI Arobas à l'Inria Sophia Antipolis. Une fois par an, nous leur présenterons ce qui a été fait et ce qu'il reste à faire pour répondre aux objectifs de l'ADT.

Hormis leur rôle d'évaluateur, ces personnes pourraient être sollicitées pour faire installer et tester ViSP dans leurs équipes respectives. Nous aurions alors la possibilité de bénéficier de *beta* testeurs en dehors de l'EPI. A l'issue de ces tests, ces utilisateurs nous feront des retours (sur les processus d'installation et de compilation, sur les points forts et les faiblesses de la documentation, sur le potentiel et la facilité d'utilisation de ViSP,...) nous permettant indéniablement d'améliorer ViSP. Un questionnaire, sous la forme d'une enquête, pourrait même être proposé aux *beta* testeurs.

Références

[1] A.H.A. Abdul Hafez, E. Cervera, and C.V. Jawahar. Optimizing image and camera trajectories in robot vision control using on-line boosting. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'07*, pages 352–357, San Diego, CA, October 2007.

[2] N. Andreff, T. Dallej, and P. Martinet. Image-based visual servoing of a gough-stewart manipulator using leg observations. *The Int. Journal of Robotics Research*, 26(7) :677–687, 2007.

[3] O. Bourquardez and F. Chaumette. Visual servoing of an airplane for alignment with respect to a runway. In *IEEE Int. Conf. on Robotics and Automation, ICRA'07*, Roma, Italia, April 2007.

[4] E. Cervera. Visual servoing toolbox for matlab / simulink. http ://vstoolbox.sourceforge.net/, 2003.

[5] E. Cervera. Cross-platform software for benchmarks on visual servoing. In *Benchmarks in Robotics Research, IROS 2006 Workshop*, Beijing, China, October 2006.

[6] A.I. Comport, E. Marchand, M. Pressigout, and F. Chaumette. Real-time markerless tracking for augmented reality : the virtual visual servoing framework. *IEEE Trans. on Visualization and Computer Graphics*, 12(4) :615–628, July 2006.

[7] P. Corke. A robotics toolbox for matlab. *IEEE Robotics and Automation Magazine*, 3(1) :24–32, September 1996.

[8] P. Corke. The machine vision toolbox. *IEEE Robotics and Automation Magazine*, 12(4), dec 2005.

[9] F.R. Correa, J.Jr. Okamoto, and M.R. Baretto. Corba-based architecture for cooperative perception. In *IFAC Workshop on Multivehicle Systems, MVS'06*, Salvador, Bahia, Brazil, October 2006.

[10] N. Courty, E. Marchand, and B. Arnaldi. Through-the-eyes control of a virtual humanoïd. In H.-S. Ko, editor, *IEEE Computer Animation, CA'01*, pages 74–83, Seoul, South Korea, November 2001.

[11] A. Crétual and F. Chaumette. Dynamic stabilization of a pan and tilt camera for submarine image visualization. *Computer Vision and Image Understanding*, 79(1) :47–65, July 2000.

[12] T. Dallej, N. Andeff, and P. Martinet. Image-based visual servoing of the i4r parallel robot without proprioceptive sensors. In *IEEE Int. Conf. on Robotics and Automation, ICRA'07*, pages 1709–1714, Roma, Italia, April 2007.

[13] T. Dallej, N. Andreff, Y Mezouar, and P. Martinet. 3d pose visual servoing relieves parallel robot control from joint sensing. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'06*, Beijing, China, October 2006.

[14] T. Dallej, H. Hadj-Abdelkader, N. Andreff, and P. Martinet. Kinematic calibration of a gough-stewart platform using an omidirectional camera. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'06*, Beijing, China, October 2006.

[15] F. Dionnet and E. Marchand. Robust stereo tracking for space robotic applications. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'07*, pages 3373–3378, San Diego, CA, October 2007.

[16] A. Diosi, A. Remazeilles, S. Segvic, and F. Chaumette. Outdoor visual path following experiments. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'07*, pages 4265–4270, San Diego, CA, October 2007.

[17] C. Doignon. An introduction to model-based pose estimation and 3d tracking techniques. In R. Stolkin, editor, *Scene Reconstruction, Pose Estimation and Tracking*. I-Tech Education and Publishing, Vienna, Austria, June 2007.

[18] G. Hager and K. Toyama. The XVision system : A general-purpose substrate for portable real-time vision applications. *Computer Vision and Image Understanding*, 69(1) :23–37, January 1998. Also Research Report Yale University.

[19] D. Kragic, V. Kyrki, P. Jensfelt, and F. Lingelbach. Drawing a parallel : Benchmarking in computer vision and robotics. In *Benchmarks in Robotics Research, IROS 2006 Workshop*, Beijing, China, October 2006.

[20] A. Krupa and F. Chaumette. Guidance of an ultrasound probe by visual servoing. *Advanced Robotics, Special issue on "Selected papers from IROS'05"*, 20(11) :1203–1218, November 2006.

[21] A. Krupa, G. Fichtinger, and G. D. Hager. Full motion tracking in ultrsound using image speckle information and visual servoing. In *IEEE Int. Conf. on Robotics and Automation, ICRA'07*, pages 2458–2464, Roma, Italia, April 2007.

[22] E. Malis, G. Morel, and F. Chaumette. Robot control from disparate multiple sensors. *Int. Journal of Robotic Research*, 20(5) :364–377, May 2001.

[23] R. Manuel and C. Santos. Evaluation of visual servoing techniques for robotic applications. Master of science thesis, KTH Computer Science and Communication, Stockholm, Sweden, 2007.

[24] E. Marchand. ViSP : A software environment for eye-in-hand visual servoing. In *IEEE Int. Conf. on Robotics and Automation, ICRA'99*, volume 4, pages 3224–3229, Detroit, Michigan, May 1999.

[25] E. Marchand. Control camera and light source positions using image gradient information. In *IEEE Int. Conf. on Robotics and Automation, ICRA'07*, pages 417–422, Roma, Italia, April 2007.

[26] E. Marchand, P. Bouthemy, F. Chaumette, and V. Moreau. Robust real-time visual tracking using a 2D-3D model-based approach. In *IEEE Int. Conf. on Computer Vision, ICCV'99*, volume 1, pages 262–268, Kerkira, Greece, September 1999.

[27] E. Marchand and F. Chaumette. Virtual visual servoing : a framework for real-time augmented reality. In G. Drettakis and H.-P. Seidel, editors, *EUROGRAPHICS'02 Conf. Proceeding*, volume 21(3) of *Computer Graphics Forum*, pages 289–298, Saarebrücken, Germany, September 2002.

[28] E. Marchand and F. Chaumette. Fitting 3d models on central catadioptric images. In *IEEE Int. Conf. on Robotics and Automation, ICRA'07*, pages 52–57, Roma, Italia, April 2007.

[29] E. Marchand, F. Chaumette, F. Spindler, and M. Perrier. Controlling an uninstrumented rov manipulator by visual servoing. In *MTS/IEEE OCEANS 2001 Conf.*, volume 3, pages 1047–1053,, Honolulu, Hawaii, novembre 2001.

[30] E. Marchand, F. Chaumette, F. Spindler, and M. Perrier. Controlling an uninstrumented manipulator by visual servoing. *The Int. Journal of Robotics Research, IJRR*, 21(7) :635–648, July 2002.

[31] E. Marchand and N. Courty. Controlling a camera in a virtual environment : Visual servoing in computer animation. *The Visual Computer Journal*, 18(1) :1–19, February 2002.

[32] E. Marchand, F. Spindler, and F. Chaumette. ViSP : A generic software platform for visual servoing. user manual. http ://www.irisa.fr/lagadic/visp/documentation/visp-manual-v0.pdf, July 2005.

[33] E. Marchand, F. Spindler, and F. Chaumette. ViSP for visual servoing : a generic software platform with a wide class of robot control skills. *IEEE Robotics and Automation Magazine*, 12(4) :40–52, December 2005. Special Issue on "Software Packages for Vision-Based Control of Motion", P. Oh, D. Burschka (Eds.).

[34] G.L. Mariottini and D. Prattichizzo. EGT for multiple view geometry and visual servoing. *IEEE Robotics and Automation Magazine*, 12(4) :26–39, December 2005.

[35] G.L. Mariottini and D. Prattichizzo. Uncalibrated video compass for mobile robots from paracatadioptric line images. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'07*, pages 226–231, San Diego, CA, October 2007.

[36] R. Mebarki, A. Krupa, and F. Chaumette. Image moments-based ultrasound visual servoing. In *IEEE Int. Conf. on Robotics and Automation, ICRA'08*, Pasadena, CA, May 2008.

[37] Y. Mezouar and F. Chaumette. Model-free optimal trajectories in the image space : Application to robot vision control. In *IEEE Int. Conference on Computer Vision and Pattern Recognition, CVPR'01*, volume 1, pages 1115–1162, Kaui, Hawaii, December 2001.

[38] M. Prats, P. Martinet, A. P. del Pobil, and S. Lee. Robotic execution of everyday tasks by means of external vision/force control. *Intelligent Service Robotics*, 2008.

[39] M. Prats, P. Martinet, del Pobil A. P., and S. Lee. Vision force control in task-oriented grasping and manipulation. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'07*, pages 1320–1325, San Diego, CA, October 2007.

[40] M. Pressigout and E. Marchand. Model-free augmented reality by virtual visual servoing. In *IAPR Int. Conf. on Pattern Recognition, ICPR'04*, volume 2, pages 887–891, Cambridge, UK, August 2004.

[41] M. Pressigout and E. Marchand. Real-time hybrid tracking using edge and texture information. *Int. Journal of Robotics Research, IJRR*, 26(7) :689–713, July 2007.

[42] F. Spindler and A. Saunier. Getting started with ViSP-2.4.0. http ://www.irisa.fr/lagadic/pdf/2007_visp-2.4.0_getting_started.pdf, February 2007.

[43] O. Tahri and F. Chaumette. Point-based and region-based image moments for visual servoing of planar objects. *IEEE Trans. on Robotics*, 21(6) :1116–1127, December 2005.

[44] T.T.H. Tran and E. Marchand. Real-time keypoints matching : application to visual servoing. In *IEEE Int. Conf. on Robotics and Automation, ICRA'07*, Roma, Italia, April 2007.

ADT

Plan du dossier de soumission

Soumission sous forme de dossier électronique (pdf et code source .tex ou .doc).

Le document de définition doit au moins contenir les éléments décrits dans ce plan, toute information que vous estimez importante à fournir dans le cadre de l'ADT peut être ajoutée au dossier.

Le dossier peut être rédigé en français ou en anglais.

Contenu du dossier du dossier de demande d'ADT.

1.1 Titre, acronyme, porteur et résumé de la proposition (1 page)

C'est la fiche signalitique.

1.2 Objectifs de l'ADT (1 à 3 pages)

Cette partie explicite la description des objectifs du développement technologique et les résultats attendus.

Le positionnement du résultat du développement doit être explicité, c'est un outil pour la recherche, c'est une contribution technologique du type logiciel libre, c'est un développement en vue d'un transfert,...
Une attention doit être portée sur l'impact attendu du résultat et sur la « vie du résultat » après l'ADT (quelle est la « sortie »).

1.3 Contexte et état de l'art (1 à 3 pages)

Positionnement du développement par rapport à l'environnement technologique de l'Inria (vis à vis du PS, des autres EPI, ...), national et international.
Les enjeux scientifiques liés au développement proposé.
Les contraintes technologiques et économiques.
Des éléments d'appréciation et de positionnement, vis-à-vis des technologies concurrentes, sur la demande utilisateur, sur les partenaires potentiels et plus généralement les éléments clé du succès et de l'impact de cette opération.

1.4 Les acteurs de l'ADT

Cette partie contient un descriptif des EPI impliquée(s), SED impliqué(s) ou d'autres partenaires. (1 page par acteur).
Si l'ADT associe des partenaires extérieurs, les raisons de ce partenariat, la contribution du/des partenaires, les aspects de propriétés des résultats seront explicités.

1.5 La description des activités de développement

Cette partie contient le plan de travail et les rôles des différentes parties impliquées avec une description des tâches, livrables, nature, niveau de maturité, qualité, plannings,..
Le nom du (des) responsable (s) technique(s).
Organisation mise en place (planning, jalons, reporting, Technical Advisory board. collège d'architecte, groupe d'utilisateurs,...).
Des informations concernant le mode de développement et les plans de qualité pourront être explicitées.
Risque technique et plan de levée des risques.

1.6 Les moyens de l'ADT

Cette partie abordera les aspects budgétaires et moyens humains (compétences des IA, chercheurs des EPI impliqués, ingénieurs des SED impliqués …) et techniques (matériels, logiciels, benchmarking, (missions), …) impliqués dans le développement.
Sources externes de financement et apports des partenaires extérieurs.

1.7 Autres points (fonction du type et le domaine de l'ADT)

Politique de protection intellectuelle et industrielle.
Politique de diffusion.
Politique de transfert.
Politique de valorisation.
Mise en place de communauté.
Actions de diffusion, de standardisation.

1.8 Évaluation

Cette partie contiendra des éléments permettant de suivre l'activité de l'ADT avec des indicateurs de succès (qualitatifs et quantitatifs).
Dans cette rubrique des noms d'experts interne et externe à l'Inria peuvent être suggérer s'ils ne sont pas impliqués dans la demande d'ADT.

Version 2 du document ;
Réf V1-PP-14 Janvier 2008