Travaux pratiques de Recalage Médical 3D
TPS TIS 3A DTMI

Ce TP a pour but de vous familiariser avec les problèmes de recalage robotique. Il sera réalisé sur plusieurs séances et il est volontairement peu guidé. Vous devrez résoudre des problèmes généraux avec peu de questions intermédiaires. Afin de limiter l'effet de problèmes pratiques délicats mais d'intérêt limité (tels que l'étalonnage des caméras endoscopiques qui nécessite beaucoup de temps), le TP est totalement réalisé en simulation. Il a été écrit de sorte à vous faire travailler simultanément sur plusieurs problèmes. Il vous paraîtra donc peut-être artificiel à certains égards. Essayez de ne pas vous focaliser sur ces possibles complications.

Il vous est demandé de rendre, à l'issue de la dernière séance, un rapport succint (maximum 10 pages, manuscrit ou dactylographié) sur le travail réalisé. Ce rapport présentera principalement les solutions de recalage que vous proposez et expliquera quels sont les problèmes à résoudre et comment vous les avez résolu. Vous essaierez de donner un regard critique à votre rapport en évoquant les limites et les adaptations pratiques nécessaires que vous entrevoyez.

1 Problème

On considère la réalisation d'une ponction à l'aide d'une aiguille rectiligne sous le guidage d'une caméra endoscopique. L'instrument (l'aiguille) est déplacé par un robot cartésien muni d'un poignet passif permettant d'assurer le passage de l'instrument par un point d'incision (voir figure 1). La caméra endoscopique est déplacée manuellement, ce qui signifie que sa position n'est pas directement contrôlée. L'objectif de la tâche médicale est de positionner automatiquement l'instrument de sorte à atteindre une cible définie manuellement par le chirurgien dans les images endoscopiques. On considère que la caméra, le robot et l'instrument ont été préalablement installés, et notamment que les trocarts sont déjà positionnés. Afin de réaliser ce geste, on dispose d'un système de localisation optique positionné dans la salle d'opération. La caméra endoscopique est munie d'un marqueur actif visible par le système de localisation. On a également attaché un marqueur à l'instrument.

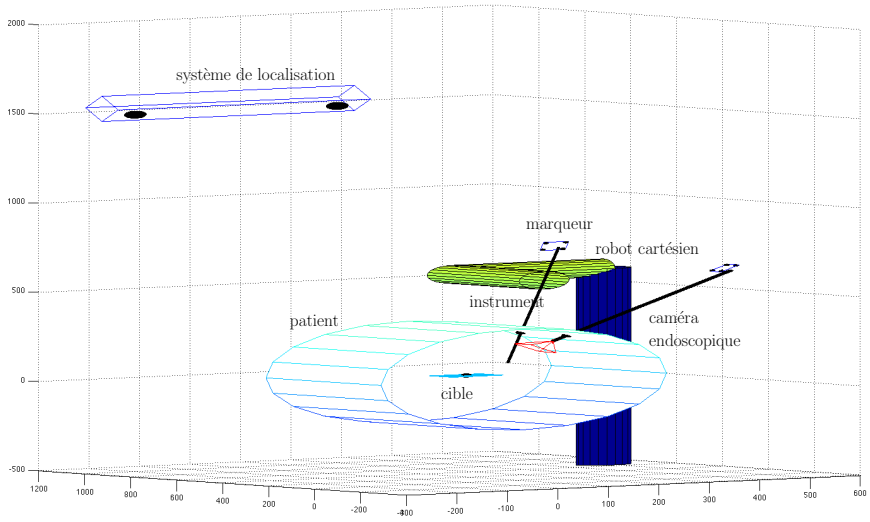

FIGURE 1 – Schéma de l'installation

1.1 Information connue

Dans le TP on considérera que les transformations suivantes sont connues :

— ${}^{cam}T_{mark\_cam}$ la transformation entre le repère de la caméra endoscopique (donc attaché à l'image) et le marqueur attaché à la caméra ; elle est définie par :
${}^{cam}R_{mark\_cam} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$ et ${}^{cam}t_{mark_cam} = \begin{pmatrix} 0 \\ -20 \\ -500 \end{pmatrix}$mm

— le Modèle Géométrique du robot ;

— la position de l'instrument par rapport à l'effecteur définie par ${}^{eff}R_{inst} = I_3$ et ${}^{eff}t_{inst} = \begin{pmatrix} 0 \\ 0 \\ 500 \end{pmatrix}$mm.

— les paramètres intrinsèques de la caméra : $K = \begin{pmatrix} 400\text{pix/mm} & 0 & 380\text{pix} \\ 0 & 400\text{pix/mm} & 285\text{pix} \\ 0 & 0 & 1 \end{pmatrix}$. On considère que la rectification due à la distorsion a déjà été réalisée.

En revanche les informations suivantes ne sont pas connues et devront donc être déterminées si elles s'avèrent nécessaires lors de la procédure de recalage :

— la position du marqueur attaché à l'instrument par rapport à l'instrument ;
— la position des trocarts (points d'incision) ;
— la position de la base du robot ;
— la position du système de localisation.

1.2 Positionnement des repères

Les position et orientation de certains repères sont prédéfinies.

Le repère lié à l'instrument $\mathcal{F}_{inst}$ est centré à l'extrémité distale de l'instrument avec w_{inst} le long de l'axe orienté de la base vers la pointe de l'instrument.

Le repère lié à l'effecteur $\mathcal{F}_{eff}$ est centré sur l'effecteur et orienté comme $\mathcal{F}_{inst}$, de sorte que ${}^{eff}R_{inst} = I_3$ et ${}^{eff}t_{inst} = (0; 0; 500)^T$mm.

Le repère lié à la caméra endoscopique $\mathcal{F}_{cam}$ est orienté de sorte que w_{cam} soit aligné avec l'axe optique, u_{cam} pointe le long de l'axe horizontal de l'image et v_{cam} le long de l'axe vertical de l'image.

2 Simulation

La simulation sera réalisée à l'aide de Matlab.

Téléchargez l'archive à l'adresse `http://eavr.u-strasbg.fr/~nageotte/TP_recalage_1920.tar.gz` et décompressez-la. Le dossier contient un ensemble de fonctions Matlab permettant de simuler le système médical et robotique servant de support au travail.

Les différents éléments (robot, caméra, système de localisation) sont gérés par une machine à états. Tous les mouvements et acquisitions sont réalisés de façon asynchrone lors de l'appel des fonctions correspondantes. L'ensemble des fonctions de lecture d'état et de changement d'état est listé ci-dessous :

1. **Fonctions de changement d'état :**

   — `InitConfig()` : place les différents éléments dans la configuration initiale de travail. Cette fonction devra être appelée à chaque fois que vous voudrez tester vos programmes afin de travailler dans une configuration répétable.
   — `MoveCamera(trans, ang)` : déplace la caméra d'un vecteur de translation `trans` exprimé dans le repère courant de la caméra et réalise une rotation autour de l'axe optique d'un angle `ang` (à donner en degrés) ; les paramètres d'entrée ne sont appliqués qu'approximativement, comme ce serait le cas pour un mouvement manuel.
   — `MoveCameraInitialPosition()` : ramène la caméra dans sa configuration initiale ; il s'agit d'une fonction de "confort" qui permet de retourner dans une configuration visuelle "utile" après des

déplacements excessifs. Cette fonction n'est pas nécessaire dans les algorithmes de résolution des problèmes posés.
— `MoveEffPosition(position)` : amène l'effecteur dans la position opérationnelle `position` définie par rapport à la base. L'orientation n'est pas imposée.
— `MoveEffVelocity(velocity, period)` : envoie au robot la vitesse opérationnelle linéaire `velocity` exprimée dans le repère de l'effecteur, pendant une durée `period` exprimée en secondes. La vitesse de rotation n'est pas imposée.

2. **Fonctions de lecture d'état :**

— `transf_base_eff = GetRobotCurrentPosition(flag)` : retourne la position de l'effecteur du robot par rapport à sa base, obtenue par le modèle géométrique direct, sous forme d'une matrice homogène 4×4.
— `mesure = GetLocalizerInformation(flag)` : retourne dans la structure mesure l'ensemble des positions des marqueurs mesurées par le système de localisation.
`mesure.mark(1).T` est la transformation entre le système de localisation et le marqueur attaché à la caméra. `mesure.mark(2).T` est la transformation entre le système de localisation et le marqueur attaché à l'instrument.
— `targ_pos = GetTargetPosition(flag)`. Cette fonction simule la sélection de la cible par l'utilisateur dans l'image courante fournie par la caméra. La position est un vecteur de dimension 2 exprimé en pixels.
— `ComputeTRE()`. Cette fonction doit être appelée à la fin de votre programme de recalage. Elle rend l'erreur de recalage à l'extrémité distale de l'instrument et vous permet donc d'évaluer si votre programme de recalage fonctionne ou non.
— `DisplayConfig()`. Affiche une représentation de la configuration courante.
Vos programmes devront appeler les fonctions précédentes. Ces fonctions ne doivent pas être modifiées et leur code n'a pas à être lu.

Dans les fonctions de lecture d'état, le paramètre d'entrée facultatif `flag` permet de tester l'effet de bruits de mesure. Dans un premier temps vous mettrez ce paramètre à 0 ou vous lancerez les fonctions correspondantes sans argument. Vous pourrez simuler l'effet du bruit dans un second temps en donnant une valeur d'entrée non nulle. L'amplitude du bruit est fixée en interne pour chaque fonction de lecture d'état concernée.

Pour écrire vos fonctions, vous disposez des fonctions de manipulation algébrique et matricielle de Matlab. N'hésitez pas à consulter l'aide Matlab ou à me demander en cas de doute. Vous avez également accès à des fonctions de manipulation des rotations fournies dans l'archive :

— `[theta,u] = r2thetau(R)` : convertit la matrice de rotation R en un axe u et l'angle theta.
— `R = thetau2r(thetau)` : convertit la rotation définie par l'axe et l'angle thetau (angle : norme de thetau, axe : vecteur thetau normé) en une matrice de rotation R.
— `R = theta2r([thetax, thetay, thetaz])` : calcul la matrice de rotation correspondant aux angles d'Euler xyz.

Remarques : il se peut que vous rencontriez des configurations dans lesquelles les programmes précédents renvoient des erreurs car je n'ai probablement pas géré l'ensemble des exceptions possibles. Dans ce cas, signalez-moi le problème pour que je le répertorie et que je le traite par la suite et prenez une autre configuration de travail.
Vous aurez peut-être envie d'avoir accès à d'autres données (par exemple les positions articulaires du robot). Si elles ne sont pas fournies c'est qu'il est possible de s'en passer. Considérez que l'impossibilité d'y accéder est une contrainte pratique.

3 Recalage avec système de localisation

Proposez, développez et validez une procédure de recalage et de positionnement permettant d'amener l'aiguille au niveau de la cible comme décrit en section 1. Pour cela vous écrirez un script et des fonctions Matlab qui feront elles-mêmes appel aux fonctions existantes de simulation décrites précédemment.

4 Recalage sans système de localisation

On choisit maintenant une autre approche basée sur l'utilisation des images endoscopiques. On supposera qu'on dispose d'un logiciel permettant d'estimer la pose de l'instrument par rapport à la caméra endoscopique sur la base d'un marqueur imprimé sur le corps de l'instrument. Cette fonction est simulée par `Tcam_inst = GetInstrumentPosition(flag)`.

— Proposez une structure pour un tel marqueur.
— Montrez que cette fonctionnalité permet théoriquement de se passer du système de localisation. Concevez un programme permettant de positionner l'instrument sans recourir à la fonction `GetLocalizerInformation(flag)`.
— Discutez des avantages et inconvénients relatifs de deux approches.

5 Evaluation de la propagation des erreurs

On souhaite évaluer l'effet des erreurs d'estimation de la position du trocart de l'instrument sur la précision de positionnement de l'instrument. Pour cela on suppose que seule la position du trocart est mal estimée en raison des erreurs de mesure de la position de l'instrument (par le système de localisation ou la mesure endoscopique selon le cas). On suppose que les autres estimations sont sans erreur.

On suppose que la variance sur la position du trocart est isotrope et d'amplitude 1 mm^2. Déterminez l'ellipsoïde représentant la variance de la position de l'extrémité distale de l'instrument en procédant de deux façons :

— évaluation en utilisant la propagation des variances ;
— évaluation numérique en simulant de nombreuses configurations bruitées.

Pour l'affichage, vous pourrez utiliser la fonction `plot_ellipsoid(center, rot_mat, axe_lengthes, scale)` qui permet de tracer un ellipsoïde centré en `center`, de demi-axes de longueurs `axes_lengthes` et orienté par la matrice de rotation `rot_mat`. `scale` permet de définir une échelle pour le tracé en cas de besoin. Le repère de référence pour le schéma est le repère "monde" (`world`). Vous devrez donc transférer les informations dans ce repère en utilisant la transformation `Tworld_base` qui est définie comme variable globale après l'utilisation de `InitConfig`.

6 Positionnement sous le contrôle de l'imageur

On souhaite s'affranchir des erreurs sur la position du trocart. Pour cela on a recours à une procédure de positionnement de l'instrument sous le contrôle visuel de l'endoscope (asservissement visuel).

— On souhaite réaliser l'asservissement sur une pose 3D reconstruite (asservissement basé position). Expliquez pourquoi ce n'est pas une bonne idée d'utiliser la pose complète obtenue avec le marqueur précédent. Quelle information reconstruite peut-on asservir ? Faites un choix et réalisez la boucle d'asservissement. Pour cela vous pourrez réaliser une boucle dans laquelle :
  1. vous calculez l'erreur de positionnement ;
  2. vous calculez un jacobien (si le jacobien est indépendant de la configuration ce calcul peut-être fait en dehors de la boucle) ;
  3. vous calculez une commande que vous envoyez au robot ;
  4. vous lisez et affichez l'état du système après réalisation du mouvement en utilisant `DisplayConfig`.
— Testez l'effet des erreurs sur la position du trocart et du bruit de mesure de la pose du marqueur. Représentez le système asservi d'un point de vue automatique et donnez les explications de ce qu'on observe en simulation.

— Pour s'affranchir encore plus des erreurs de recalage, la technique usuelle consiste à utiliser un asservissement basé image où l'erreur est directement exprimée dans le repère de l'image. Expliquez pourquoi ce n'est pas possible pour l'application considérée ici.