ÉCOLE UNIVERSITAIRE
DE PHYSIQUE ET D'INGÉNIERIE
Université Clermont Auvergne

UNIVERSITÉ CLERMONT AUVERGNE

Clermont-Ferrand

École Universitaire de Physique et d'Ingénierie

RAPPORT DE TRAVAIL D'ÉTUDE ET DE RECHERCHE

Master Automatique et Robotique

1re année de Master

Idir Messali

Mise en service de robots sous ROS 2 et mise en place d'une expérimentation d'observation humaine

Date de soutenance :	10/07/2026
Entreprise/Laboratoire :	Institut Pascal
Encadrant :	Sébastien Lengagne
Année universitaire :	2025–2026

Remerciements

Je tiens à remercier Monsieur Sébastien Lengagne pour son encadrement, ses conseils techniques et scientifiques, ainsi que pour l'opportunité de travailler sur un sujet concret associant robotique, ROS 2, vision par ordinateur, structuration de données expérimentales et préparation à l'apprentissage humain–robot.

Je remercie également les personnes ayant contribué à l'accès au matériel robotique, à la compréhension de l'environnement logiciel et aux essais réalisés pendant le projet. Le travail présenté dans ce rapport a été mené avec l'objectif de fournir une base utile aux futurs étudiants, stagiaires, doctorants et chercheurs amenés à reprendre ou à prolonger ces développements.

Enfin, je remercie également Rigon Ajvazi et Jérôme Nortier, avec qui j'ai partagé le laboratoire pendant ces deux mois de projet. Les échanges au quotidien ont contribué à créer un environnement de travail agréable et stimulant. Je remercie aussi les personnes ayant participé aux essais de collecte et de validation intermédiaire, qui ont permis de vérifier le fonctionnement de l'interface, du protocole de segmentation et du pipeline de traitement dans des conditions proches d'une utilisation expérimentale réelle.

Résumé

Ce Travail d'Étude et de Recherche porte sur la mise en service de robots sous ROS 2 et la mise en place d'une expérimentation d'observation humaine. Il vise à préparer une base technique et expérimentale permettant d'étudier le lien entre des manipulations réalisées par un opérateur humain et leur réutilisation dans un contexte robotique. Le travail réalisé a permis de rendre opérationnelle une cellule robotique de manipulation et de construire une plateforme capable d'accompagner une expérimentation humain–cube, depuis l'enregistrement des actions jusqu'à la production de données exploitables. Les résultats obtenus constituent ainsi un support pour de futurs travaux sur l'analyse de gestes, la reconstruction de trajectoires et le transfert de mouvements humains vers des robots manipulateurs.

Mots-clés : ROS 2, robot manipulateur, UR3e, observation humaine, plateforme expérimentale, manipulation humain–cube, reconstruction de trajectoires, transfert humain–robot.

Abstract

This Research Study Project focuses on the commissioning of robots under ROS 2 and the development of an experimental setup for human observation. It aims to prepare a technical and experimental basis for studying the link between manipulations performed by a human operator and their reuse in a robotic context. The work carried out made it possible to make a robotic manipulation cell operational and to build a platform supporting a human–cube experiment, from the recording of actions to the production of exploitable data. The obtained results provide a basis for future work on gesture analysis, trajectory reconstruction and the transfer of human movements to robotic manipulators.

Keywords : ROS 2, robotic manipulator, UR3e, human observation, experimental platform, human–cube manipulation, trajectory reconstruction, human–robot transfer.

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

ROS	Robot Operating System
ROS 2	Robot Operating System 2
TER	Travail d'Étude et de Recherche
UR3e	Universal Robots 3e
IHM	Interface Homme-Machine
TCP	Tool Center Point
RTDE	Real-Time Data Exchange
ArUco	Augmented Reality University of Cordoba
MediaPipe	MediaPipe Framework
CSV	Comma-Separated Values
rosbag	ROS bag
RViz	ROS Visualization
UNN	Universal Notice Network
LS-UNN	Latent-Space Universal Notice Network

Introduction générale

La robotique expérimentale nécessite une articulation cohérente entre matériel, logiciel, protocole de test et organisation des données. Un robot manipulateur disponible dans un laboratoire doit être configuré, relié à son environnement logiciel, commandable depuis un poste opérateur et documenté pour pouvoir être repris par d'autres utilisateurs. De la même manière, une expérimentation d'observation humaine doit produire des données structurées, traçables et exploitables pour l'analyse ou pour des travaux de transfert humain–robot.

Le sujet de ce Travail d'Étude et de Recherche, intitulé *Mise en service de robots sous ROS 2 et mise en place d'une expérimentation d'observation humaine*, s'inscrit dans cette logique. Il comporte deux volets complémentaires. Le premier concerne la remise en service d'une cellule robotique UR3e sous ROS 2, avec l'intégration du robot, de la pince, de la caméra et d'une interface opérateur. Le second porte sur la mise en place d'une plateforme de collecte humain–cube destinée à produire un dataset organisé pour des travaux liés au transfert humain–robot.

Le travail réalisé vise à rendre des outils expérimentaux plus accessibles, plus structurés et plus facilement réutilisables. Pour la partie robotique, l'objectif est de disposer d'une cellule UR3e commandable sous ROS 2, avec ses périphériques intégrés et une documentation permettant la reprise du système. Pour la partie observation humaine, l'objectif est de proposer un protocole de collecte reproductible, une interface web de pilotage, une segmentation automatique des manipulations et une chaîne de traitement offline produisant des fichiers exploitables.

Le rapport est organisé en quatre chapitres. Le premier présente le cadre scientifique, les objectifs du TER et l'analyse de l'existant. Le deuxième détaille la remise en service de la cellule UR3e sous ROS 2. Le troisième décrit la plateforme de collecte humain–cube, le protocole expérimental et l'organisation du dataset. Le quatrième présente le traitement offline, la reconstruction des poses et des trajectoires, les sorties générées et leur validation sous RViz.

Chapitre 1

Positionnement du TER dans la chaîne de transfert humain–robot

1.1 Chaîne de transfert humain–robot

Le projet s'inscrit dans le cadre du transfert de gestes humains vers des robots manipulateurs. Dans ce domaine, une démonstration humaine est utilisée comme source d'information pour apprendre, adapter ou valider une tâche robotique. Cette logique se rapproche de l'apprentissage par démonstration, où le robot exploite des exemples d'exécution fournis par un humain pour construire une politique de commande ou une représentation de la tâche [11].

Dans le cas étudié, la démonstration correspond à une manipulation simple d'un cube, dans la continuité des travaux précédents sur la création d'une base de mouvements humains pour le transfert vers des robots manipulateurs [2]. Chaque manipulation est décrite par un ensemble de variables expérimentales : mouvement de la main, pose du cube, horodatage des mesures et état final de l'objet. L'association de ces variables permet d'analyser le lien entre l'action humaine observée et l'effet produit dans l'espace de travail. Ces données constituent l'entrée nécessaire aux méthodes de transfert humain–robot, qu'il s'agisse d'un modèle direct [14] ou d'une architecture de type UNN/LS-UNN [12, 13]. La Figure 1.1 illustre cette première étape du projet : les manipulations humaines réalisées sur le cube sont collectées afin d'alimenter une base de données exploitable.

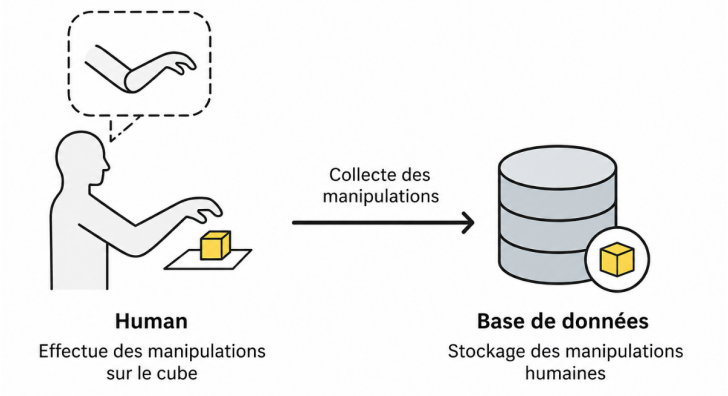


FIGURE 1.1 – Principe de collecte des manipulations humaines pour la constitution d'une base de données.

La Figure 1.2 présente la chaîne générale du projet. La base de données occupe une position centrale : elle relie l'observation humaine aux modèles capables de produire ensuite une sortie robotique exploitable.

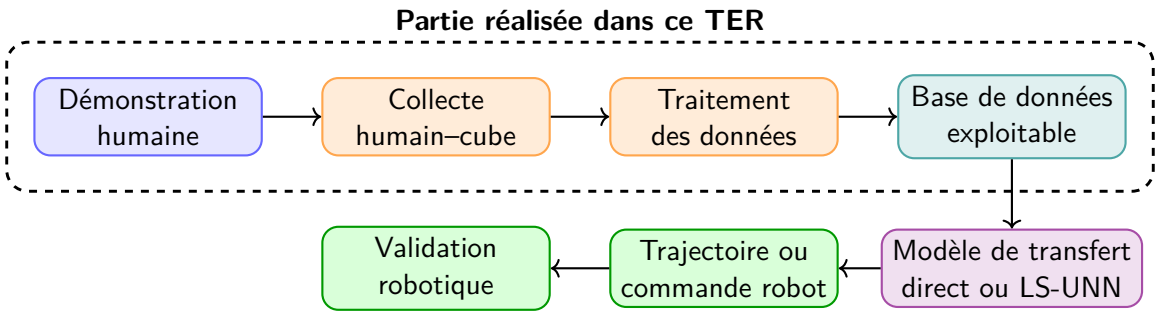


FIGURE 1.2 – Chaîne globale du transfert humain–robot avec indication de la partie réalisée dans ce TER.

Deux familles de méthodes sont particulièrement liées à cette chaîne. La première est le modèle direct : il consiste à apprendre une relation entre des observations issues de l'humain et de l'objet, puis une sortie utilisable pour un robot. Cette idée est proche des approches de transfert direct, où une information issue du geste humain est transformée en action ou en représentation exploitable par un robot manipulateur [14].

La seconde famille est l'UNN et sa variante LS-UNN. L'UNN cherche à séparer la connaissance propre à la tâche des éléments dépendants du robot, afin de faciliter le transfert entre agents différents [12]. Le LS-UNN prolonge cette logique en introduisant un espace latent commun appris à partir de trajectoires appariées et alignées temporellement [13]. Dans ce cas, l'humain et le robot ne sont pas comparés directement articulation par articulation : leurs données sont projetées dans un espace commun permettant de traiter des morphologies différentes [13].

Ces méthodes exigent une description expérimentale complète de chaque manipulation. Pour chaque séquence, la base doit contenir la pose du cube, le mouvement humain, l'horodatage des mesures, les limites de début et de fin de manipulation, ainsi que les métadonnées de session. Ces éléments donnent au modèle de transfert les informations nécessaires pour relier une action humaine observée à une sortie robotique exploitable.

1.2 Équipe de travail au laboratoire

Pendant la durée du TER, le travail a été réalisé dans un environnement partagé avec d'autres stagiaires du laboratoire. Jérôme Nortier travaillait sur la mise en place d'un espace de représentation pour le transfert humain–robot. Ce travail est directement lié à l'exploitation future des données produites par la plateforme de collecte humain–cube présentée dans ce rapport.

Rigon Ajavzi travaillait, quant à lui, sur la détection et l'estimation 3D d'une balle rapide à l'aide d'une caméra événementielle. La présence de ces deux stagiaires dans le même environnement de travail a permis de partager le laboratoire, les échanges techniques et le contexte expérimental pendant les deux mois du projet.

1.3 Travaux précédents sur la collecte humain–cube

Le travail de Gabriel de Souza Marcos constitue le point de départ expérimental de cette continuité. Son stage portait sur la création d'une base de données de mouvements humains pour le transfert vers des robots manipulateurs [2]. Il a étudié la capture du mouvement humain avec MediaPipe et le suivi du cube avec ArUco, afin d'associer le geste humain à la trajectoire de l'objet manipulé.

Ce travail a permis de valider une première chaîne de collecte humain–cube. Il a montré qu'il était possible de produire des données utiles pour le transfert humain–robot : mouvement du corps et des mains, pose du cube, informations temporelles et premières sorties exploitables. Il a aussi proposé une première segmentation automatique des séquences, basée sur la détection de la main gauche.

L'analyse de ce travail montre que les choix techniques étaient cohérents et adaptés au problème. MediaPipe permet d'extraire des informations sur le mouvement humain, tandis qu'ArUco fournit une estimation de pose simple et robuste pour le cube. Le travail de Gabriel constitue donc une base de faisabilité solide, mais certains points devaient être renforcés pour faciliter la reprise du projet et l'exploitation future des données.

Dans ce TER, les améliorations apportées concernent surtout la structuration générale de la collecte et sa reproductibilité. Les principaux apports peuvent être résumés ainsi :

- simplifier l'utilisation de la plateforme de collecte pour un utilisateur non expert ;
- améliorer le moyen de séparation des manipulations afin de rendre l'expérience plus contrôlable ;
- renforcer la traçabilité des données collectées, notamment en conservant les vidéos brutes comme référence ;
- mieux organiser le passage entre la collecte et le traitement des données ;
- préparer des sorties exploitables pour l'analyse, la visualisation et les futures validations robotiques.

Cette continuité permet de passer d'une première chaîne expérimentale à une plateforme plus structurée, plus facile à utiliser et mieux adaptée à l'enrichissement progressif d'une base de données pour les futurs travaux de transfert humain–robot.

1.4 Données nécessaires aux modèles de transfert

Les modèles de transfert humain–robot exploitent des données qui décrivent simultanément l'action, l'objet et le temps. Pour une tâche de manipulation, la trajectoire du cube représente l'effet produit dans l'environnement, tandis que les points du bras et de la main décrivent l'action humaine qui a conduit à ce résultat. Cette double information est importante pour passer d'une observation humaine à une représentation utilisable par un robot.

Dans un modèle direct, les données peuvent être utilisées pour apprendre une correspondance entre les variables observées et une sortie robotique. Dans une approche LS-UNN, elles servent plutôt à construire ou exploiter un espace de représentation commun entre agents. Les deux approches ont donc besoin d'un dataset cohérent, où chaque manipulation est identifiable, datée, retraitable et reliée à ses fichiers sources.

La base produite par ce TER doit notamment servir à un travail parallèle portant sur la mise en place d'un espace de représentation pour le transfert humain–robot. Elle fournit les séquences humain–cube nécessaires pour tester des méthodes comme le modèle direct ou le LS-UNN. La mise en service des robots sous ROS 2 complète ce besoin en préparant un environnement où les sorties du modèle pourront ensuite être visualisées, validées ou testées.

La figure 1.3 illustre cette utilisation du dataset. La base de données issue des manipulations humain–cube peut alimenter un modèle direct, qui apprend une relation entre deux agents, ou une approche LS-UNN, qui passe par un espace latent commun.

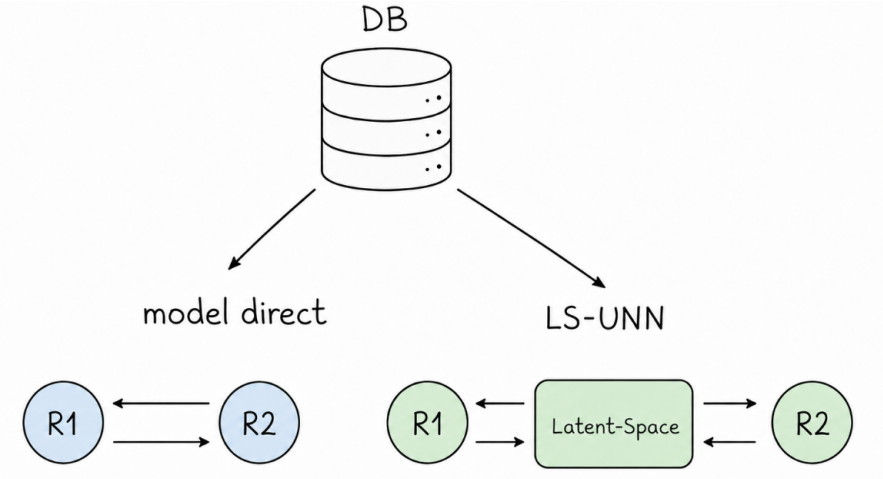


FIGURE 1.3 – Utilisation de la base de données pour deux approches de transfert : modèle direct et LS-UNN.

1.5 Contributions retenues dans le TER

Le sujet officiel du TER demande deux actions principales : remettre en service des robots existants sous ROS 2 et mettre en place une expérimentation d'observation d'activités humaines [1]. Dans le périmètre réalisé, ces deux actions deviennent deux contributions complémentaires.

La première contribution concerne la cellule UR3e. Les trajectoires produites par les futurs modèles de transfert doivent pouvoir être testées sur un robot réel ou dans un environnement proche de l'exploitation robotique. Pour cela, la cellule UR3e doit être accessible sous ROS 2, commandable depuis un PC et accompagnée d'une procédure de lancement claire. Le travail réalisé comprend l'intégration du driver ROS 2, l'établissement de la communication avec le contrôleur du robot, la commande de l'organe terminal, la récupération du flux caméra et la création d'une interface opérateur. Cette mise en service fournit une plateforme robotique utilisable pour vérifier, visualiser et préparer les essais issus des données de transfert humain–robot.

Le Niryo Ned2 n'a pas été traité comme une plateforme à porter entièrement sous ROS 2, car des paquets existent déjà pour ce robot. Il intervient dans ce TER comme support de validation préparatoire : les trajectoires reconstruites peuvent être visualisées dans RViz avec le modèle du robot et préparées pour des essais ultérieurs.

La seconde contribution concerne la plateforme d'observation humain–cube. Elle couvre la collecte vidéo, la détection ArUco, l'extraction MediaPipe, la segmentation automatique des manipulations, l'organisation du dataset et la génération de sorties exploitables. Les fichiers produits doivent pouvoir être utilisés par les travaux de transfert, mais aussi par les outils ROS 2 comme les rosbags et RViz.

La Figure 1.4 situe précisément la place du TER. La partie encadrée correspond à mon intervention : produire la base de données expérimentale et préparer l'environnement robotique de validation.

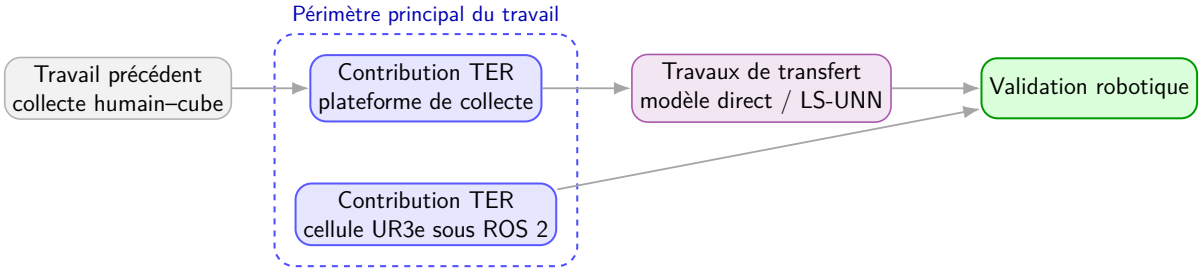


FIGURE 1.4 – Positionnement du TER dans le projet global.

1.6 Analyse de l'existant et besoins identifiés

L'analyse de départ a porté sur deux éléments : les outils de collecte issus des travaux précédents et les plateformes robotiques disponibles au laboratoire. Le Tableau 1.1 résume les limites identifiées et les réponses apportées dans le TER.

TABLE 1.1 – Analyse de l'existant et réponses apportées dans le cadre du TER.

Élément analysé	**Limite identifiée et réponse apportée**
Première collecte humain–cube	La reprise était difficile sans protocole de session suffisamment structuré. Le besoin était de stabiliser la collecte et de faciliter son utilisation par d'autres personnes. La réponse apportée repose sur l'interface web, la gestion des sessions et les fichiers de métadonnées.
Base vidéo des manipulations	Les données nécessaires au transfert doivent être produites, enregistrées et conservées comme source expérimentale. La réponse apportée repose sur l'enregistrement des vidéos brutes, l'organisation par sessions et le stockage des manipulations collectées.
Exploitation des vidéos collectées	Les vidéos collectées ne fournissent pas directement les trajectoires ni les poses utilisables par les modèles. La réponse apportée repose sur la détection ArUco, l'extraction MediaPipe et le traitement offline.
Découpage des gestes	Les limites de début et de fin doivent être identifiées de manière stable pendant la collecte. La réponse apportée repose sur l'utilisation du marqueur ArUco `ID99` comme déclencheur de segmentation.
Robots du laboratoire	Les environnements sont hétérogènes entre ROS 1, ROS 2 et certaines configurations anciennes. La réponse apportée repose sur la mise en service de la cellule UR3e et la centralisation du lancement.
Travaux de transfert humain–robot	Les modèles nécessitent des données alignées, identifiables et retraitables. La réponse apportée repose sur la génération de fichiers CSV, rosbags, trajectoires et visualisations RViz.

Cette analyse a conduit à séparer le travail en deux axes. Le premier axe traite l'environnement robotique, afin de rendre l'UR3e exploitable sous ROS 2. Le second axe traite la chaîne de collecte et de traitement, afin de fournir les données nécessaires aux modèles de transfert.

1.7 Cahier des charges fonctionnel

Le cahier des charges retenu traduit le sujet initial en fonctions vérifiables. Il traduit les objectifs généraux en éléments concrets permettant d'évaluer le travail réalisé. Le Tableau 1.2 présente ces besoins et leurs critères de réussite.

Table 1.2 – Cahier des charges fonctionnel du TER.

Besoin	**Réponse proposée**	**Critère de réussite**
Disposer d'une cellule robotique exploitable	UR3e sous ROS 2	- Robot connectable, topics et services disponibles
Commander les périphériques	Intégration pince et caméra	- Pince commandable, flux caméra affichable
Simplifier l'utilisation du robot	Interface opérateur	-Lancement centralisé et supervision des états principaux
Collecter des manipulations humain–cube	Interface web de collecte	- Session contrôlable par un utilisateur non expert ROS 2
Séparer les manipulations	Déclenchement par marqueur ArUco `ID99`	- Début et fin détectés sans découpage manuel
Conserver la source expérimentale	Vidéos brutes et métadonnées	- Chaque manipulation identifiable et retraitable
Produire des données pour les modèles	Traitement offline	- CSV, rosbags et trajectoires générés
Préparer la validation robotique	RViz, Niryo Ned2 et UR3e	- Trajectoires visualisables et prêtes pour essais futurs

Le chapitre suivant détaille le premier axe du TER : la remise en service de la cellule UR3e sous ROS 2 et l'intégration de ses périphériques.

Chapitre 2

Mise en service de la cellule UR3e sous ROS 2

2.1 Support matériel de départ

La cellule robotique utilisée dans ce TER est construite autour d'un bras collaboratif Universal Robots UR3e, appartenant à la gamme e-Series d'Universal Robots [3]. Le robot est monté sur une table rainurée et associé à son contrôleur Universal Robots ainsi qu'au teach pendant, utilisé pour l'allumage, le déverrouillage, le contrôle manuel et les opérations côté PolyScope.

Le bras UR3e est équipé d'une pince Robotiq Hand-E et d'une Robotiq Wrist Camera, toutes deux fixées sur l'organe terminal. La pince constitue l'élément de préhension, tandis que la caméra de poignet fournit une image de la zone proche de l'effecteur. L'ensemble est piloté depuis un PC sous Ubuntu 22.04 avec ROS 2 Humble, relié au contrôleur UR3e par un câble Ethernet. La Figure 2.1 présente la cellule UR3e disponible au laboratoire avec ses principaux éléments matériels.

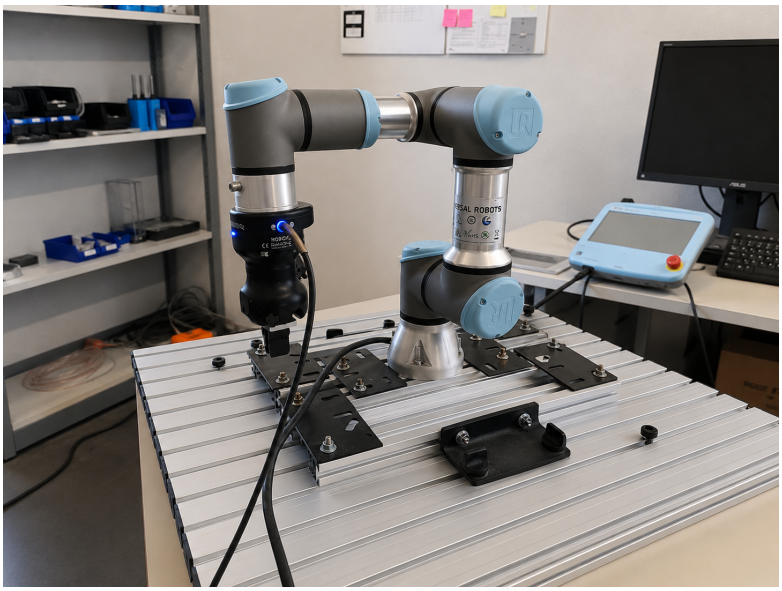


FIGURE 2.1 – Cellule UR3e disponible au laboratoire avec pince Robotiq Hand-E et Robotiq Wrist Camera.

2.2 Point de départ et objectif de la mise en service

Le point de départ du travail est une cellule UR3e disponible physiquement au laboratoire, mais restée inactive depuis plusieurs années. Le robot a été récupéré cette année afin d'être remis en service dans un environnement ROS 2 complet. Les travaux précédents et les configurations anciennes fournissaient des éléments utiles, mais la reprise nécessitait une chaîne cohérente autour de ROS 2 Humble, du driver du robot, des périphériques et d'une procédure de lancement claire.

Le besoin est lié à la suite du projet de transfert humain–robot. Les sorties produites par les futurs modèles de transfert doivent pouvoir être vérifiées sur un robot réel ou dans un environnement proche de l'exploitation robotique. La cellule UR3e doit donc être connectable depuis un PC ROS 2, commandable, documentée et utilisable sans imposer à chaque utilisateur de relancer manuellement tous les composants.

Les objectifs retenus pour cette mise en service sont les suivants :

— rendre le robot UR3e accessible sous ROS 2 ;
— rendre l'organe terminal commandable ;
— rendre le flux caméra disponible dans ROS 2 ;
— centraliser le lancement des composants principaux ;
— fournir une interface opérateur pour superviser et commander la cellule ;
— préparer une plateforme de validation pour les trajectoires issues de la base de données humain–cube ;
— produire une documentation claire afin de faciliter la reprise de la cellule et les prochains travaux réalisés sur ce robot.

La démarche suivie est résumée dans la Figure 2.2. Elle part du matériel disponible et se termine par une validation fonctionnelle de la cellule.

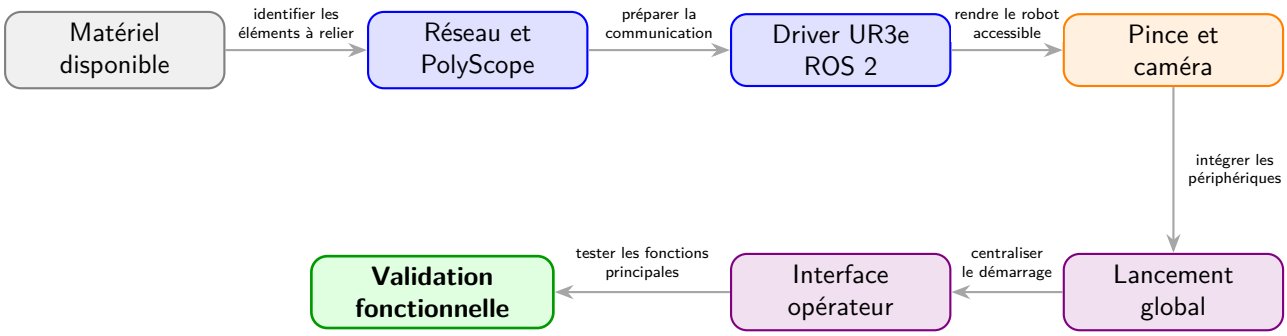


FIGURE 2.2 – Démarche générale de mise en service de la cellule UR3e sous ROS 2.

2.3 Chaîne de communication du robot UR3e

La cellule UR3e devait être exploitée dans un environnement ROS 2, alors que le contexte initial pouvait conduire à une migration d'une configuration existante sous ROS 1 vers ROS 2. L'analyse des solutions disponibles a montré qu'un driver Universal Robots compatible ROS 2 existait déjà. Le choix technique a donc été d'utiliser ce driver comme base de communication, au lieu de reconstruire manuellement toute la couche d'accès au robot à partir de l'ancien environnement ROS 1.

Le driver retenu est le *Universal Robots ROS 2 Driver*, disponible dans le dépôt officiel `Universal_Robots_ROS2_Driver` [5, 6]. Ce driver assure la liaison entre le PC opérateur et le contrôleur UR3e. Il permet de récupérer les états articulaires du robot, d'accéder aux interfaces de commande et de transmettre les consignes de trajectoire vers la cellule réelle.

La communication avec le robot repose aussi sur la configuration côté contrôleur. Le PC est relié au contrôleur UR3e par un câble Ethernet, tandis que le programme *External Control*, lancé depuis PolyScope, autorise la prise de contrôle par un PC externe. Côté ROS 2, le driver s'appuie sur la couche `ros2_control` et sur les contrôleurs de trajectoire pour faire le lien entre les commandes ROS 2 et le bras robotique [7].

La chaîne de communication entre le PC opérateur et le robot réel est représentée dans la Figure 2.3. Elle montre les couches principales impliquées entre l'environnement ROS 2 et le bras UR3e.

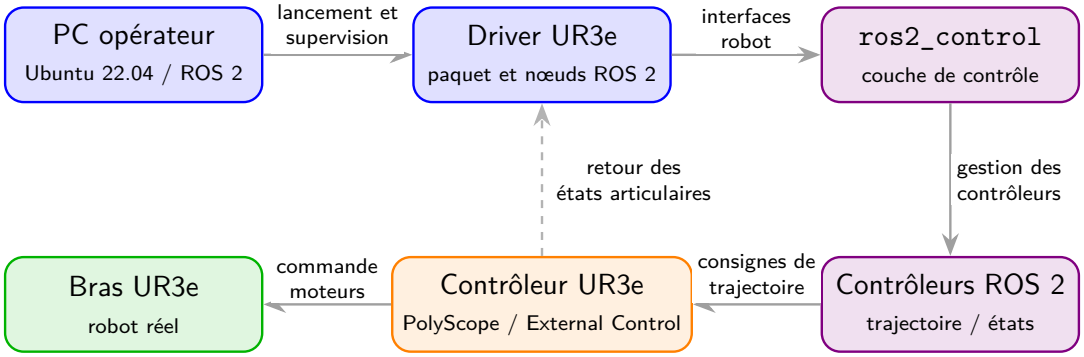


FIGURE 2.3 – Chaîne de communication entre le PC ROS 2 et le robot UR3e réel.

La validation de cette chaîne repose sur des critères fonctionnels précis. Les états articulaires doivent être publiés dans ROS 2, les contrôleurs doivent être actifs et une consigne de mouvement doit pouvoir être envoyée puis exécutée sur le robot réel dans des conditions de sécurité.

2.4 Intégration de l'ensemble terminal

2.4.1 Intégration de la pince Robotiq Hand-E

La pince Robotiq Hand-E [9] ajoute une capacité de préhension à la cellule UR3e. Son intégration dans ROS 2 consiste à rendre les commandes d'ouverture et de fermeture accessibles depuis le PC opérateur, sans manipuler directement la communication bas niveau avec le périphérique.

La solution retenue repose sur le package `ros2_RobotiqGripper`, disponible dans le dépôt `IFRA-Cranfield/ros2_RobotiqGripper` [8]. Ce package fournit un serveur ROS 2 qui expose la commande de la pince sous forme de service. Une fois lancé, la pince devient accessible à travers le service `/Robotiq_Gripper`. Les ordres d'ouverture et de fermeture peuvent alors être envoyés sous forme de requêtes simples, comme `OPEN` et `CLOSE`. Cette abstraction permet de transformer une commande matérielle spécifique en une interface ROS 2 plus simple à tester et à intégrer dans l'interface opérateur.

Le principe retenu est présenté dans la Figure 2.4. Une commande issue de ROS 2 ou de l'interface opérateur est transmise au service dédié, puis relayée vers la pince à travers la communication avec le contrôleur du robot.

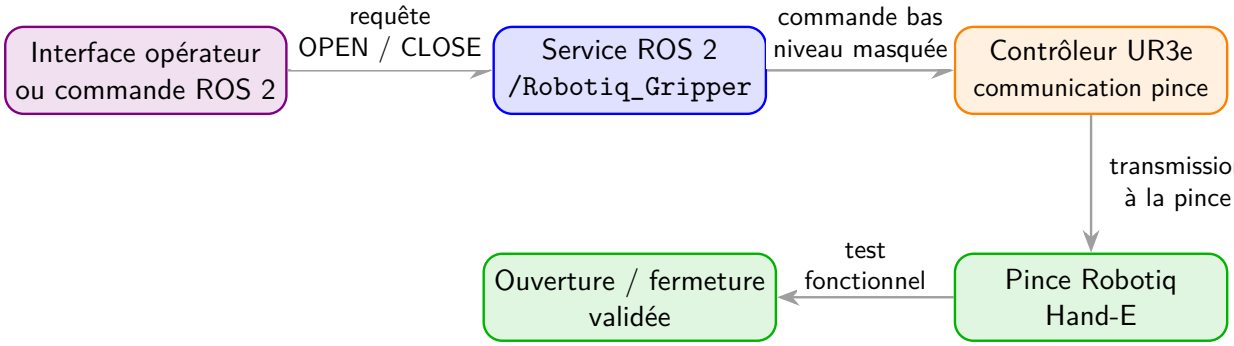


FIGURE 2.4 – Principe d'intégration de la pince Robotiq Hand-E dans la cellule ROS 2.

L'intégration de la pince se résume donc par trois éléments : l'utilisation d'un service ROS 2 dédié, l'intégration de ce service dans le lancement global de la cellule et la validation par l'exécution réelle des commandes d'ouverture et de fermeture.

2.4.2 Intégration du flux de la Robotiq Wrist Camera

La Robotiq Wrist Camera est fixée sur l'organe terminal du robot. Elle fournit un retour visuel proche de la pince, utile pour observer la zone située devant l'effecteur pendant l'utilisation de la cellule.

Le flux image de la Robotiq Wrist Camera est récupéré depuis son interface réseau. conformément au principe d'accès aux images fourni par la documentation Robotiq [10]. L'intégration repose donc sur un bridge logiciel, chargé de lire l'image fournie par la caméra puis de la publier dans ROS 2 sous forme de topic image. Dans la cellule, ce rôle est assuré par le package `robotiq_camera_bridge`, qui publie le flux sur le topic `/robotiq_camera/image_raw`. Cette solution transforme un flux caméra spécifique en une donnée standard exploitable par les outils ROS 2 et par l'interface opérateur.

Le principe d'intégration est présenté dans la Figure 2.5. La caméra fournit l'image au bridge, le bridge publie cette image dans ROS 2, puis l'interface opérateur s'abonne au topic pour afficher le retour visuel.

FIGURE 2.5 – Principe d'intégration du flux de la Robotiq Wrist Camera dans la cellule ROS 2.

La validation de cette intégration consiste à vérifier que le topic image est bien publié dans ROS 2 et que le flux de la caméra est affiché dans l'interface opérateur. Cette vérification confirme que le retour visuel de l'ensemble terminal est disponible pendant l'utilisation de la cellule.

2.5 Architecture générale de la cellule et interface opérateur

L'architecture générale de la cellule UR3e a été organisée autour d'un fichier de lancement global. Le fichier `ur3e_full_operator.launch.py` regroupe les composants nécessaires au fonctionnement de la cellule : le driver du robot, les contrôleurs ROS 2, le service de commande de la pince, le bridge caméra et l'interface opérateur. Ce choix permet de démarrer la cellule à partir d'un point d'entrée unique, au lieu de lancer séparément chaque composant depuis plusieurs terminaux.

Les principaux composants lancés dans cette architecture sont résumés dans le Tableau 2.1.

TABLE 2.1 – Composants principaux lancés par `ur3e_full_operator.launch.py`.

Package	**Composant lancé**	**Rôle dans la cellule**
`ur_robot_driver`	Inclusion de `ur_control.launch.py`	Démarrage du driver UR3e, de la communication avec le contrôleur, des contrôleurs ROS 2 et des fonctions de supervision du robot.
`ros2_robotiqgripper`	`server.py`	Lancement du serveur de commande de la pince Robotiq Hand-E et exposition du service `/Robotiq_Gripper`.
`robotiq_camera_bridge`	`robotiq_camera_bridge`	Récupération de l'image caméra par HTTP, puis publication du flux dans ROS 2 sur le topic `/robotiq_camera/image_raw`.
`ur3e_operator_interface`	`operator_interface`	Lancement de l'interface opérateur et création des nœuds `/ur3e_motion_controller` et `/ur3e_camera_viewer`.

La Figure 2.6 présente l'organisation générale de la cellule. Le fichier de lancement global démarre les composants techniques principaux, puis l'interface opérateur s'appuie sur ces composants pour centraliser l'accès à la cellule.

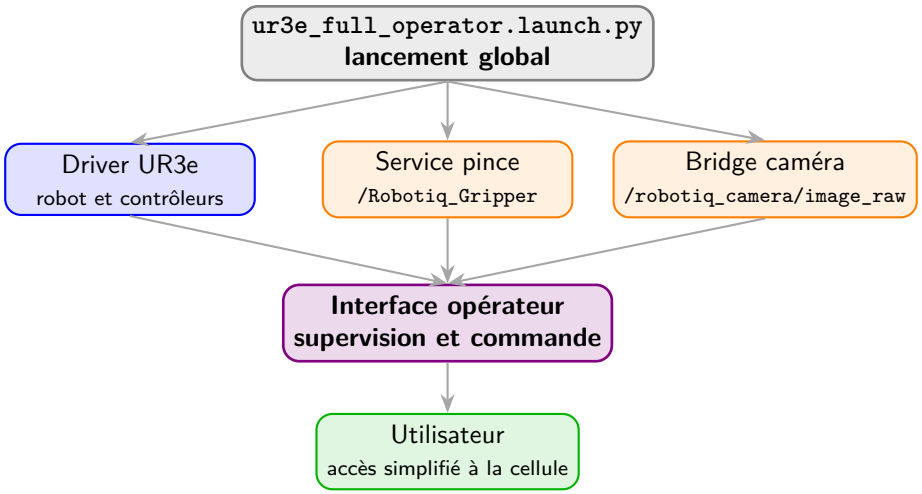


FIGURE 2.6 – Organisation générale de la cellule UR3e autour du lancement global et de l'interface opérateur.

L'interface opérateur constitue une couche d'accès au-dessus des composants lancés par l'architecture globale. Elle utilise le driver du robot, le service de pince et le bridge caméra

pour regrouper dans un même outil les fonctions utiles pendant les essais : affichage du flux caméra, lecture des états du robot, commande de la pince, préparation de mouvements et accès à certaines commandes de supervision.

L'interface développée est une interface graphique locale de type IHM, exécutée sur le PC opérateur. Elle est lancée comme un programme Python associé à l'environnement ROS 2 de la cellule. Techniquement, elle est développée dans le package `ur3e_operator_interface`. Elle utilise `rclpy`, la bibliothèque Python permettant d'écrire des nœuds ROS 2, Tkinter pour la construction de l'interface graphique, ainsi que Pillow pour le traitement et l'adaptation des images avant leur affichage. Pour l'affichage caméra, l'image publiée dans ROS 2 est récupérée, convertie dans un format compatible avec Tkinter, puis affichée dans l'interface.

La Figure 2.7 présente l'interface opérateur développée pour la cellule UR3e.

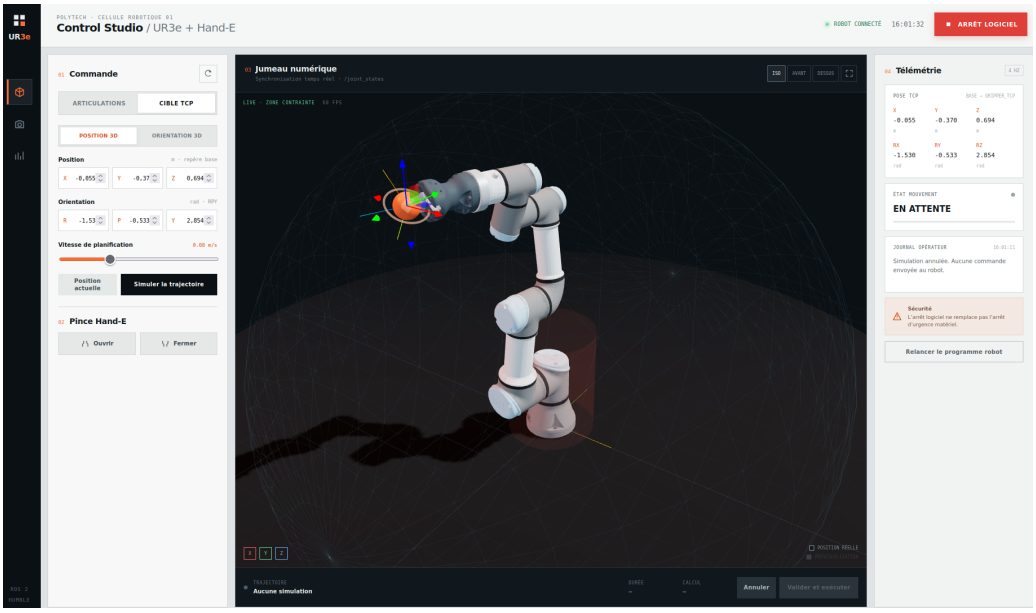


FIGURE 2.7 – Interface opérateur développée pour la supervision et la commande de la cellule UR3e.

La structure interne de l'interface opérateur est résumée dans le Tableau 2.2.

TABLE 2.2 – Structure interne de l'interface opérateur.

Bloc fonctionnel	**Interface ROS 2 utilisée**	**Fonction principale**
Affichage caméra	`/robotiq_camera/image_raw`	Réception du flux caméra et affichage dans l'interface Tkinter.
Supervision du robot	`/joint_states` `/tcp_pose_broadcaster/pose`	Lecture des états du robot et mise à jour des informations affichées dans l'interface.
Commande de la pince	`/Robotiq_Gripper`	Envoi des ordres d'ouverture et de fermeture de la pince Robotiq Hand-E.
Trajectoires articulaires	`/scaled_joint_trajectory_controller/follow_joint_trajectory`	Envoi de trajectoires articulaires au contrôleur du robot.
Commandes URScript	`/urscript_interface/script_command`	Transmission de commandes directes pour certains modes de déplacement.
Relance du programme robot	`/dashboard_client/play` `/io_and_status_controller/resend_robot_program`	Relance contrôlée du programme robot côté PolyScope.

Cette organisation rend la cellule plus simple à démarrer, à superviser et à manipuler pendant les essais. Le lancement global assure le démarrage cohérent des composants, tandis que l'interface opérateur fournit un accès direct aux fonctions principales de la cellule.

2.6 Validation fonctionnelle

La validation fonctionnelle vérifie que les composants intégrés répondent aux besoins définis au début du chapitre. Les tests portent sur la communication avec le robot, la disponibilité des états, l'activation des contrôleurs, la commande de l'organe terminal, l'affichage caméra, l'interface opérateur et la préparation des mouvements.

Le Tableau 2.3 synthétise les validations réalisées.

TABLE 2.3 – Synthèse de validation fonctionnelle de la cellule UR3e.

Élément validé	**Vérification réalisée**	**Résultat obtenu**	**État**
Chaîne UR3e sous ROS 2	Communication PC–robot, états articulaires et contrôleurs	Robot accessible et contrôleurs actifs	✓
Pince Robotiq Hand-E	Commandes d'ouverture et de fermeture	Commandes exécutées depuis ROS 2 et l'interface	✓
Flux caméra	Publication et affichage de l'image	Flux disponible dans ROS 2 et visible dans l'interface	✓
Interface opérateur	Regroupement des fonctions principales	Supervision et commandes centralisées	✓
Mouvement robot	Préparation et envoi d'une trajectoire	Trajectoire exécutable après validation utilisateur	✓
Manuel technique	Documentation de la cellule	Manuel rédigé pour faciliter la reprise	✓

Les validations réalisées montrent que la cellule UR3e est exploitable sous ROS 2 comme support de test. Le robot, l'organe terminal, la caméra et l'interface opérateur sont intégrés dans une chaîne cohérente. Cette mise en service prépare les essais futurs liés aux trajectoires issues de la base de données humain–cube.

2.7 Sécurité lors des essais sur robot réel

Les essais réalisés sur la cellule UR3e impliquent un robot réel en mouvement. La validation fonctionnelle est associée aux conditions de sécurité mises en place pendant les tests. Les mouvements ont été préparés progressivement, avec une vitesse réduite et une vérification de la zone de travail avant l'exécution sur le robot.

La sécurité repose principalement sur le contrôle de l'environnement de test, la disponibilité de l'arrêt d'urgence physique et la validation préalable des commandes envoyées au robot. Cette démarche permet de limiter les risques liés aux déplacements du bras, à la présence de l'organe terminal et à l'exécution de trajectoires depuis ROS 2.

Ainsi, les tests de communication, de commande de la pince, d'affichage caméra et de mouvement robot ont été réalisés uniquement lorsque les conditions matérielles et logicielles permettaient une exécution contrôlée sur la cellule réelle.

Chapitre 3

Plateforme de collecte humain–cube et organisation du dataset

3.1 Objectif de la plateforme de collecte

Cette partie du TER concerne la mise en place d'une plateforme de collecte humain–cube destinée à produire une base de données exploitable pour des travaux de transfert humain–robot. Cette base doit notamment servir au travail de Jérôme Nortier, présenté au chapitre précédent, portant sur la mise en place d'un espace de représentation pour le transfert humain–robot. Dans ce contexte, la plateforme doit fournir des données organisées, traçables et suffisamment structurées pour être reprises par un stagiaire, un doctorant ou un futur utilisateur travaillant sur ces méthodes.

Dans ce chapitre, une tâche désigne une consigne de manipulation donnée à un opérateur humain, par exemple déplacer un cube d'une zone vers une autre ou empiler un cube sur un autre. Le transfert humain–robot désigne l'exploitation de démonstrations humaines afin de produire des représentations, des trajectoires ou des données d'apprentissage utilisables ensuite par un robot manipulateur. Les approches comme UNN ou LS-UNN s'inscrivent dans cette logique, car elles cherchent à représenter une tâche dans un espace commun entre différents agents [12, 13].

La contribution présentée dans ce chapitre porte donc sur la partie expérimentale amont : définir une tâche de manipulation, rendre la collecte accessible, enregistrer les données, segmenter automatiquement les manipulations et organiser les résultats sous une forme exploitable. La Figure 3.1 présente la logique générale de cette contribution.

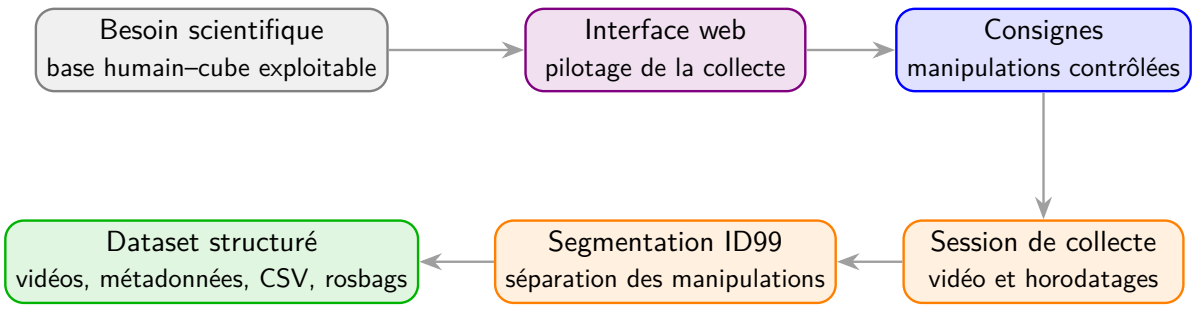


FIGURE 3.1 – Vue générale de la plateforme de collecte humain–cube.

La plateforme proposée répond à trois besoins du cahier des charges : rendre le protocole de collecte reproductible, enregistrer les manipulations réalisées par les participants et organiser automatiquement les données produites. L'interface web a été ajoutée pour rendre la collecte accessible à un utilisateur qui ne connaît pas nécessairement ROS 2 ou la structure interne du projet.

3.2 Principe général de la tâche humain–cube

La tâche expérimentale retenue consiste à demander à un opérateur humain de réaliser des manipulations simples avec des cubes placés sur une zone de travail. Chaque manipulation est associée à une consigne explicite. Cette consigne indique l'action attendue, par exemple déplacer un cube vers une zone donnée, empiler un cube sur un autre ou modifier une configuration déjà présente sur la table.

Le dispositif expérimental est organisé autour d'une table de manipulation. Cette table contient des zones de dépôt, les cubes manipulés et des marqueurs ArUco utilisés comme repères visuels. Les zones servent à formuler les consignes de manière claire, tandis que les marqueurs permettent de relier les enregistrements vidéo aux traitements réalisés ensuite. La Figure 3.2 présente l'organisation de la table expérimentale ainsi qu'une vue réelle de l'espace de travail utilisé pendant la collecte.

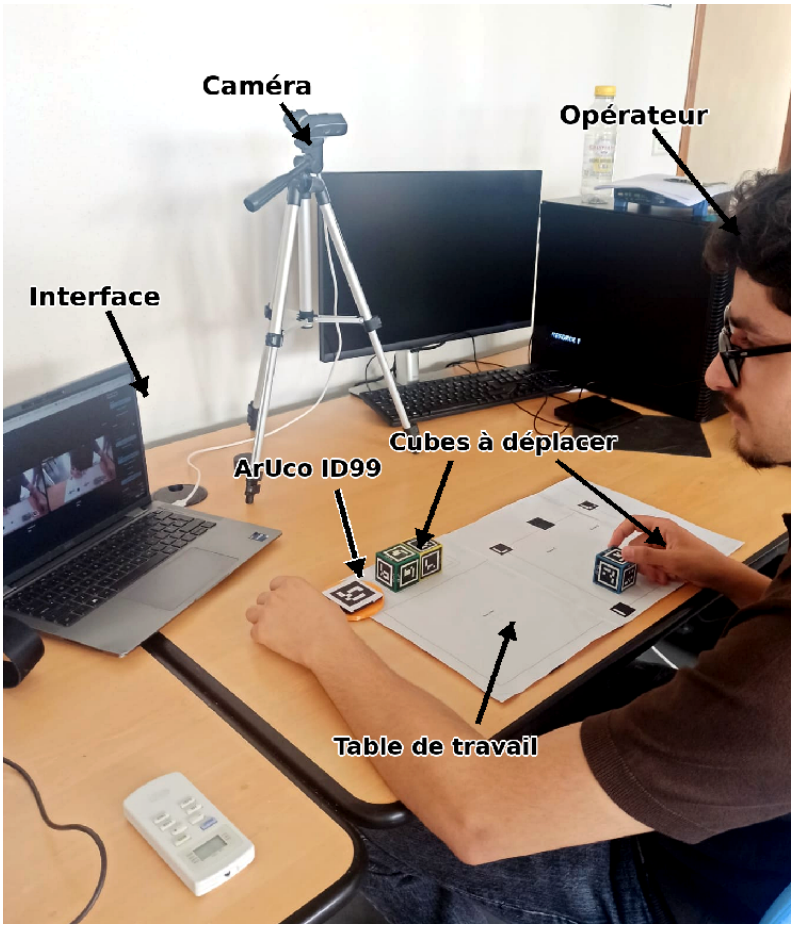


FIGURE 3.2 – Organisation de la table expérimentale et vue réelle de l'espace de travail utilisé pour la collecte humain–cube.

Les tâches de manipulation retenues sont inspirées du travail de Gabriel de Souza Marcos sur la création d'une base de mouvements humains pour le transfert vers des robots manipulateurs [2]. Dans cette continuité, le choix expérimental consiste à conserver, pour chaque manipulation, les informations nécessaires à l'exploitation ultérieure : la consigne, la vidéo, les horodatages, les métadonnées de session et les fichiers générés par les traitements.

L'hypothèse de travail retenue consiste à relier chaque action humaine à une évolution observable de la scène. Cette hypothèse guide l'organisation du dataset : chaque manipulation doit pouvoir être retrouvée à partir de sa consigne, de sa séquence vidéo et des données produites par les traitements offline.

La Figure 3.3 résume le principe de la tâche utilisée dans la plateforme.

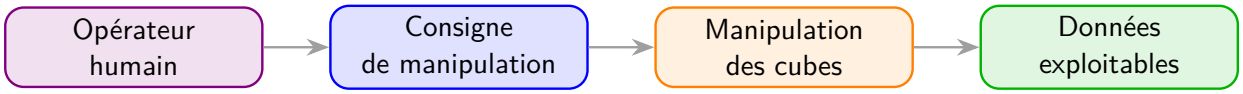


FIGURE 3.3 – Principe général de la tâche humain–cube.

Ce principe permet de relier le déroulement expérimental à la structure du dataset : une manipulation correspond à une consigne, à une séquence enregistrée et à des données exploitables pour les traitements suivants.

3.3 Génération des consignes et diversité des manipulations

Avant le début d'une session, l'utilisateur renseigne les paramètres de collecte, notamment l'identifiant de l'opérateur et le nombre de manipulations à réaliser. À partir de ces paramètres, la plateforme génère un plan de consignes. Ce plan définit l'ordre des manipulations demandées pendant la session.

Les consignes correspondent à des actions simples sur les cubes. Elles peuvent demander un déplacement entre deux zones, un empilement, un désassemblage ou une réorganisation de la scène. Cette diversité enrichit la base de données en variant les positions de départ, les positions d'arrivée, les directions de déplacement et les relations entre les cubes.

La génération des consignes comporte une part aléatoire contrôlée. La plateforme conserve une représentation interne de l'état supposé des cubes afin de produire des demandes cohérentes. Par exemple, elle évite de demander un déplacement vers la même zone ou de déplacer un cube utilisé comme support par un autre cube. Les consignes restent ainsi variées tout en restant compatibles avec l'état courant de la scène.

Le Tableau 3.1 donne des exemples de consignes pouvant être générées pendant une session.

TABLE 3.1 – Exemples de consignes générées pendant une session de collecte.

Type de manipulation	**Exemple de consigne**
Déplacement simple	Déplacer le cube rouge de la zone A vers la zone C.
Empilement	Placer le cube bleu sur le cube rouge.
Désassemblage	Retirer le cube bleu du dessus du cube rouge et le placer en zone B.
Réorganisation	Déplacer le cube vert vers une zone libre afin de modifier la configuration de la scène.

3.4 Principe de segmentation des manipulations

La collecte est organisée en manipulations successives. Pour séparer automatiquement ces manipulations, la plateforme utilise un marqueur ArUco spécifique, identifié par l'`ID99`. Ce choix d'identifiant est arbitraire, le marqueur `ID99` a été retenu comme repère de déclenchement, sans propriété particulière liée à sa valeur. Il sert de signal temporel pendant la session, sa détection pendant une durée définie déclenche le début d'un segment, tandis que sa disparition pendant une durée définie clôture ce segment.

Cette segmentation remplace une découpe manuelle après l'acquisition. Chaque segment obtenu correspond à une manipulation associée à une consigne, à une séquence vidéo et à un dossier de données. L'ID99 joue donc un rôle central dans l'organisation du dataset, car il permet de séparer les manipulations dès la phase de collecte.

La Figure 3.4 présente le principe de segmentation automatique utilisé pendant une session.

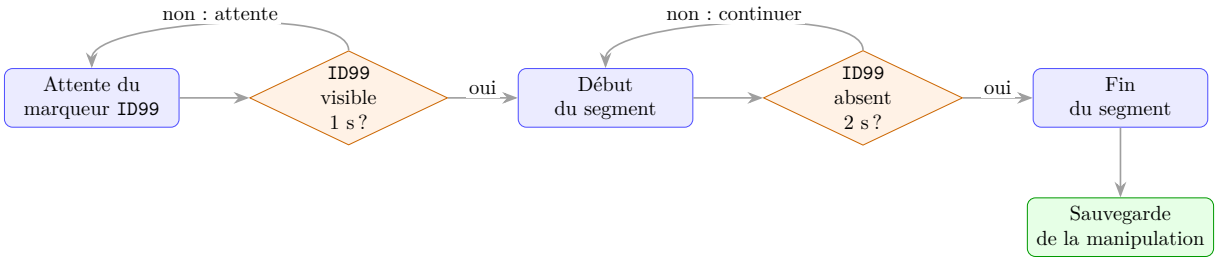


FIGURE 3.4 – Principe de segmentation automatique d'une manipulation par le marqueur `ID99`.

3.5 Interface web de collecte

L'interface web de collecte a été ajoutée au cours du TER comme amélioration pratique de la plateforme expérimentale. Elle ne constituait pas une exigence initiale du sujet, mais elle a été développée afin de faciliter l'exécution du protocole, la récupération des données et la reprise de la plateforme par d'autres utilisateurs. Cette interface regroupe dans un outil accessible depuis un navigateur les actions nécessaires à la collecte, préparation d'une session, génération des consignes, suivi de l'avancement et organisation des fichiers produits.

Elle permet à un utilisateur de lancer une session sans manipuler directement les commandes ROS 2 ni l'organisation interne du projet. L'opérateur renseigne l'identifiant du participant et le nombre de manipulations à réaliser, puis l'interface génère les consignes, affiche la consigne courante et suit l'avancement de la collecte.

L'intérêt de cette interface est de rendre possible la répétition du protocole avec plusieurs participants, plusieurs sessions et plusieurs configurations de cubes. Cette répétition contrôlée permet d'augmenter progressivement la taille du dataset tout en conservant une organisation homogène des données.

La Figure 3.5 présente l'interface web utilisée pour piloter la collecte humain–cube.

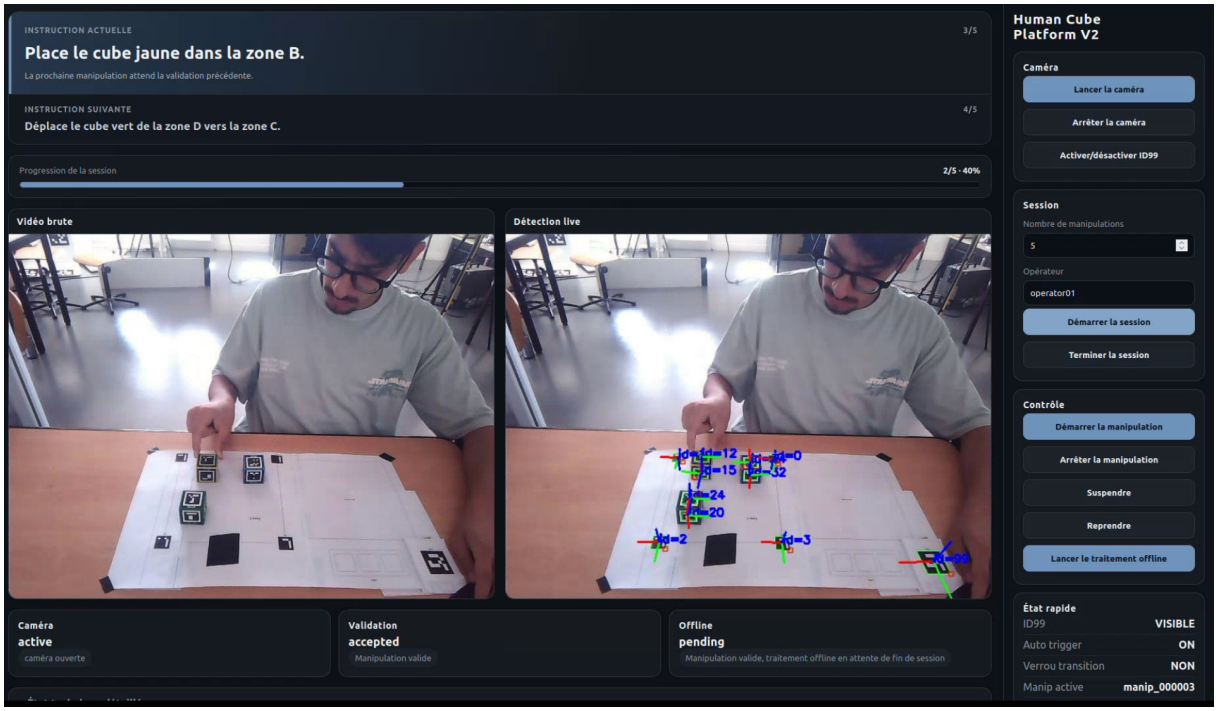


FIGURE 3.5 – Interface web utilisée pour le pilotage de la collecte humain–cube.

Pendant une session, l'interface assure le lien entre la consigne demandée, l'enregistrement vidéo et l'organisation des données produites. Les données collectées servent ensuite d'entrée à la chaîne de traitement offline présentée dans le chapitre suivant.

3.6 Déroulement d'une session de collecte

Une session de collecte regroupe plusieurs manipulations réalisées par un même opérateur. Elle commence par la création d'une session dans l'interface web, puis par la génération du plan de consignes. L'utilisateur suit ensuite les consignes affichées, réalise les manipulations demandées et utilise le marqueur ID99 pour séparer les séquences.

La Figure 3.6 présente le déroulement général d'une session de collecte. Le schéma est volontairement simplifié afin de rester lisible à l'impression.

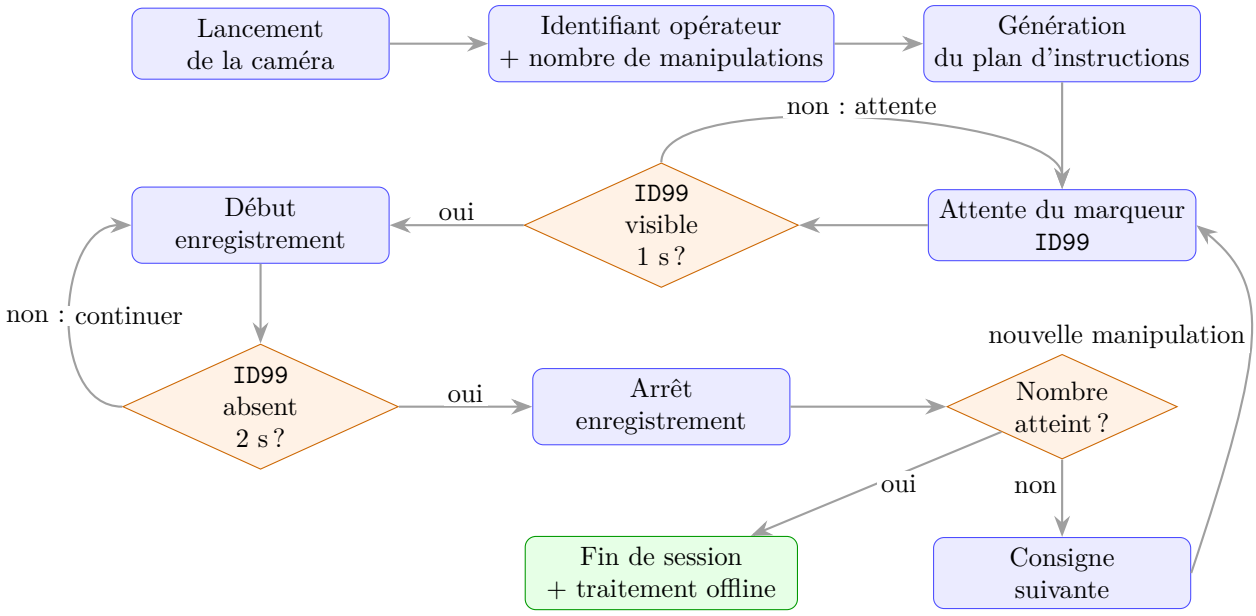


FIGURE 3.6 – Déroulement général d'une session de collecte.

Cette organisation permet d'enchaîner plusieurs manipulations dans une même session. L'utilisateur garde un protocole clair, tandis que la plateforme conserve les informations nécessaires pour relier chaque manipulation à sa consigne et à ses données enregistrées.

3.7 Données enregistrées et organisation du dataset

La collecte produit une organisation hiérarchique simple : un dossier principal regroupe l'ensemble du dataset, chaque session possède son propre dossier, puis chaque session contient les manipulations réalisées par l'opérateur. Cette structure permet de relier clairement une session, une consigne et les données enregistrées pour chaque manipulation.

Chaque manipulation contient les éléments nécessaires à son exploitation : instruction, métadonnées, vidéo brute, horodatages et fichiers générés par les traitements. Les sorties de type CSV et rosbag sont produites pour faciliter l'analyse, la visualisation et la réutilisation des données dans un environnement ROS 2 [22].

Le Tableau 3.2 résume le rôle des principaux éléments enregistrés dans le dataset.

TABLE 3.2 – Organisation des données produites par la plateforme de collecte.

Élément du dataset	**Rôle**
Dossier de session	- Regroupe les manipulations réalisées par un opérateur pendant une même session.
Plan de consignes	- Conserve l'ordre des actions demandées pendant la session.
Dossier de manipulation	- Associe une consigne à une séquence vidéo et aux fichiers générés pour cette manipulation.
Vidéo brute et horodatages	- Constituent la référence expérimentale utilisée pour les traitements offline.
Fichiers CSV	- Stockent les données numériques extraites ou reconstruites sous forme tabulaire.
Rosbag reconstruit	- Permet de rejouer les données dans un environnement ROS 2 et de les exploiter avec les outils associés.

Chapitre 4

Traitement offline, reconstruction des trajectoires et validation

4.1 Objectif du traitement offline

Le chapitre précédent a présenté la plateforme de collecte, l'organisation des sessions et la structure du dataset. Ce chapitre présente la chaîne de traitement offline appliquée aux données collectées. L'entrée principale de cette chaîne est la vidéo brute d'une manipulation, accompagnée de ses horodatages, de sa consigne et de ses métadonnées. Les sorties attendues sont des fichiers structurés permettant d'analyser les manipulations, de reconstruire les poses des cubes, de suivre certaines informations du mouvement humain et de rejouer les résultats dans un environnement ROS 2.

La Figure 4.1 présente la logique générale du traitement offline sous forme entrée–traitement–sorties.

FIGURE 4.1 – Principe général du traitement offline : entrées, traitements et sorties.

Le traitement offline a donc pour rôle de transformer les enregistrements produits pendant la collecte en données exploitables. Il reprend les vidéos enregistrées, reconstruit les informations utiles image par image, puis génère des fichiers adaptés à l'analyse, à la visualisation et à la réutilisation dans ROS 2.

4.2 Chaîne de traitement à partir des données collectées

La chaîne de traitement est appliquée à chaque manipulation enregistrée. Elle part du dossier de manipulation produit par la collecte, puis applique plusieurs étapes : lecture de la vidéo brute, détection des marqueurs ArUco, estimation des poses des cubes, extraction des points humains avec MediaPipe, synchronisation temporelle, filtrage et génération des fichiers de sortie.

La Figure 4.2 présente cette chaîne sous une forme synthétique.

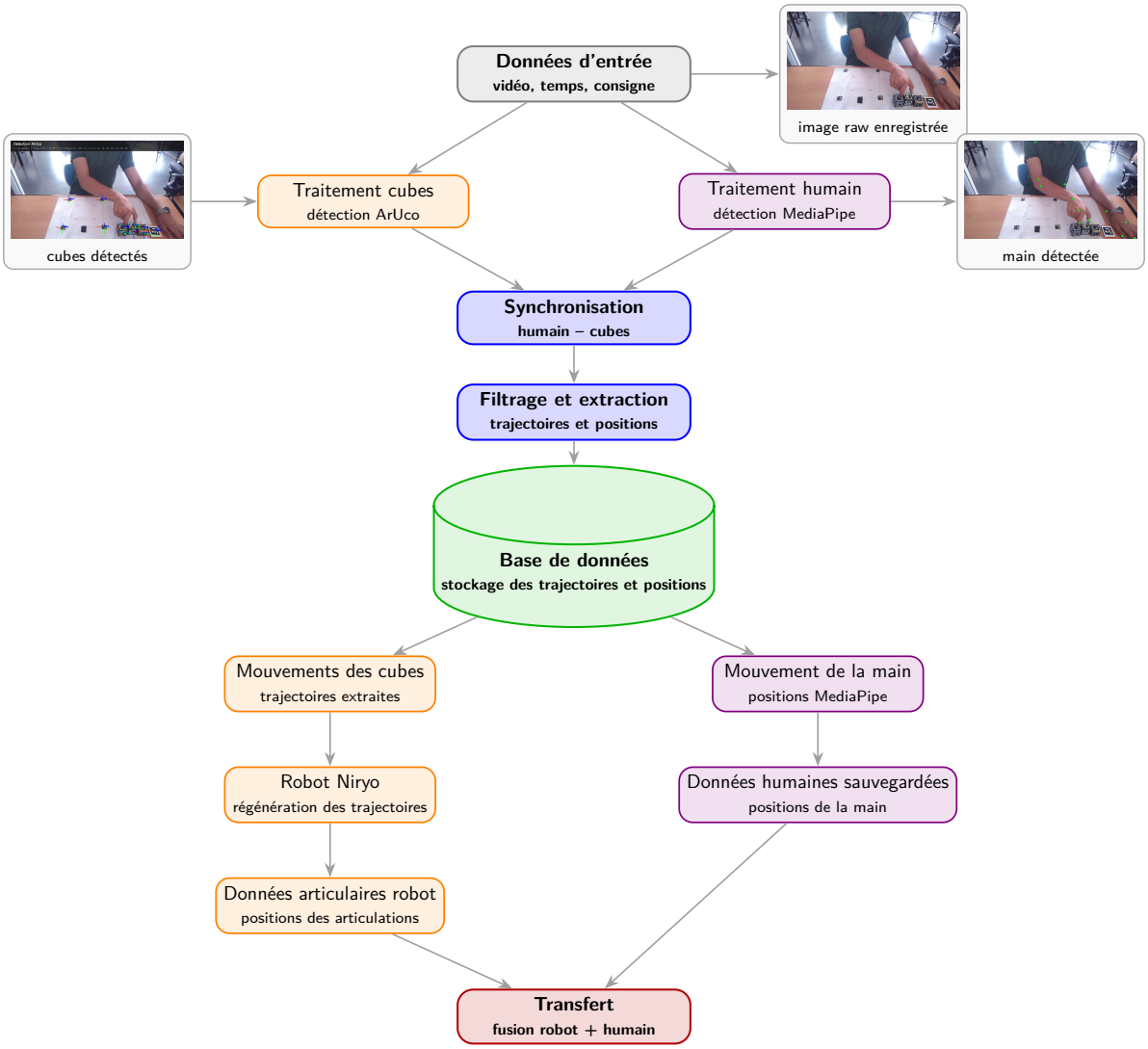


FIGURE 4.2 – Chaîne de traitement offline appliquée aux données collectées.

Les étapes principales sont résumées dans le Tableau 4.1.

TABLE 4.1 – Étapes principales de la chaîne de traitement offline.

Étape	**Données utilisées**	**Résultat produit**
Lecture de la vidéo	Vidéo brute et horodatages	Images indexées temporellement pour la manipulation.
Détection ArUco	Images RGB et calibration caméra	Marqueurs détectés, poses des repères et informations de table.
Reconstruction des cubes	Marqueurs placés sur les faces des cubes	Estimation des poses des cubes dans le repère de travail.
Extraction humaine	Images RGB de la manipulation	Points caractéristiques du corps, du bras et des mains.
Filtrage et correction	Trajectoires brutes et séries temporelles	Trajectoires plus régulières et fichiers corrigés.
Export structuré	Résultats calculés et métadonnées	CSV, rapports JSON, rosbag reconstruit et sorties de visualisation.

Cette organisation sépare clairement la collecte et l'exploitation des données. La collecte conserve les sources expérimentales, tandis que le traitement offline produit des fichiers dérivés pouvant être vérifiés, régénérés et comparés.

4.3 Détection des marqueurs ArUco et reconstruction des poses des cubes

La reconstruction des cubes s'appuie sur les marqueurs ArUco visibles dans les vidéos enregistrées. Ces marqueurs fiduciaires sont adaptés à la détection et à l'estimation de pose à partir d'images caméra [15, 16]. Chaque image de la vidéo est analysée afin de détecter les marqueurs présents sur la table ou sur les cubes. Pour chaque marqueur détecté, le traitement estime une position et une orientation à partir de ses coins dans l'image et des paramètres de calibration disponibles.

La pose estimée est d'abord obtenue dans le repère de la caméra. Elle est ensuite exprimée dans le repère de travail défini pour la table expérimentale. Ce repère de travail sert de référence commune pour représenter les zones, les cubes et les trajectoires. Les sorties associées à cette étape contiennent donc des poses datées, exprimées dans un repère spatial cohérent avec la scène de manipulation.

La Figure 4.3 présente le principe de reconstruction des poses des cubes à partir des marqueurs ArUco.

FIGURE 4.3 – Principe de reconstruction des poses des cubes à partir des marqueurs ArUco.

La Figure 4.4 illustre un exemple d'application de cette étape sur une image extraite de la vidéo. On y observe la scène de manipulation avec les cubes détectés à partir de leurs marqueurs ArUco.

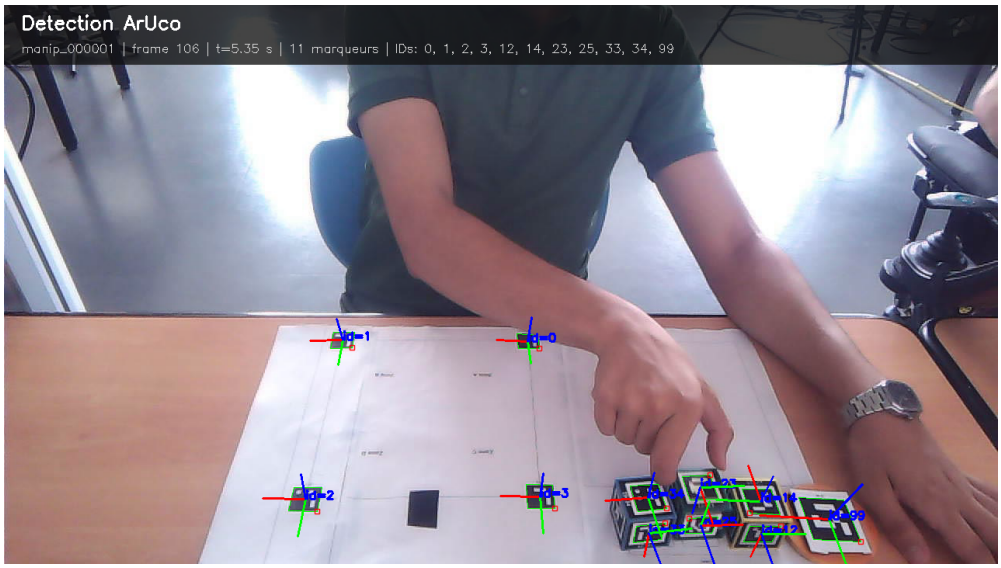


FIGURE 4.4 – Exemple de détection des cubes par marqueurs ArUco sur une image extraite de la vidéo de manipulation.

Les fichiers produits par cette étape contiennent notamment l'identifiant du cube ou du marqueur, l'horodatage, la position 3D et l'orientation. La position est représentée par les coordonnées `x`, `y` et `z`. L'orientation peut être stockée sous forme de quaternion `qx`, `qy`, `qz` et `qw`. Cette représentation est adaptée à la visualisation 3D et aux outils ROS 2.

4.4 Extraction du mouvement humain avec MediaPipe

Le traitement offline utilise MediaPipe pour extraire des points caractéristiques du mouvement humain à partir des images de la vidéo [17]. Cette étape permet d'associer à chaque image une description temporelle du geste réalisé par l'opérateur pendant la manipulation.

Dans la chaîne de traitement, les points conservés sont choisis en fonction de leur intérêt pour le transfert vers un robot manipulateur.Les points pris en compte sont : l'épaule droite, le coude droit, le poignet droit, le pouce droit, l'index droit, le point MCP du majeur droit, la hanche gauche et la hanche droite. Les points du corps sont issus du suivi de pose, tandis que les points de la main sont liés au suivi de la main proposé dans MediaPipe [18, 19]. Ces points décrivent principalement le bras droit et la main droite, qui portent l'action de manipulation, ainsi qu'un repère corporel fourni par les deux hanches.

Ce choix permet de représenter le geste humain sous une forme compatible avec une interprétation robotique. L'épaule, le coude et le poignet droits donnent une description simplifiée du bras humain, comparable à une chaîne articulée de robot manipulateur. Les points de la main droite, notamment le pouce, l'index et le MCP du majeur, fournissent une information sur la zone de saisie. Ils peuvent être interprétés comme une approximation de l'organe terminal humain, en lien avec une pince robotique utilisée pour des tâches de type *pick-and-place*. Les hanches gauche et droite servent de repères supplémentaires pour stabiliser l'interprétation de la posture globale de l'opérateur.

Pour chaque point détecté, les coordonnées sont associées à l'indice de l'image et à l'horodatage correspondant. Cette indexation permet de synchroniser le mouvement humain avec les poses des cubes reconstruites à partir des marqueurs ArUco. Les sorties MediaPipe sont donc stockées sous forme de séries temporelles dans les fichiers CSV dédiés aux landmarks humains.

La Figure 4.5 présente un exemple d'application du traitement MediaPipe sur une image extraite de la vidéo. Les points caractéristiques détectés permettent de suivre le mouvement du bras droit et de la main droite de l'opérateur pendant la manipulation.

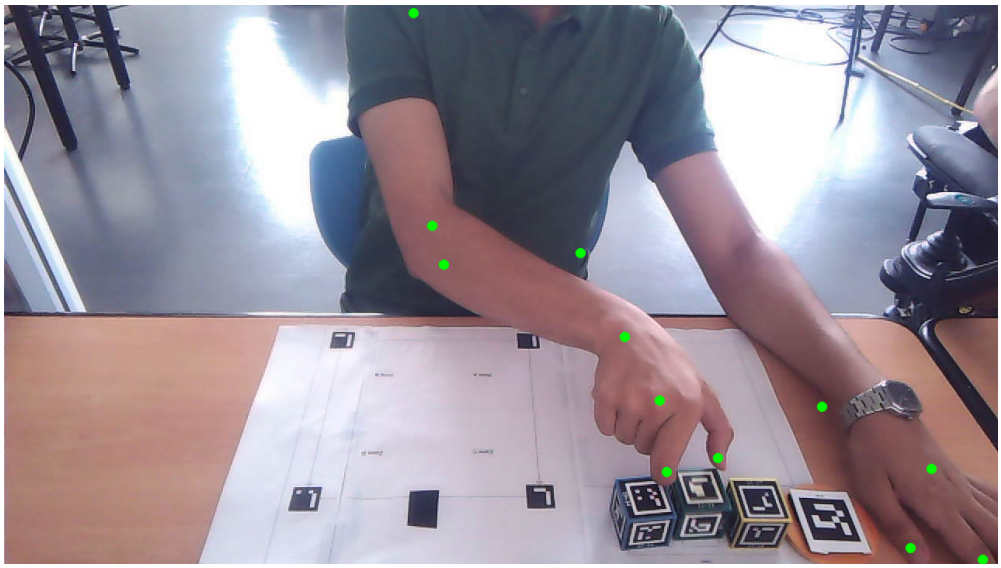


FIGURE 4.5 – Exemple d'extraction du mouvement humain avec MediaPipe sur une image extraite de la vidéo de manipulation.

4.5 Synchronisation, filtrage et correction des trajectoires

Les données extraites par ArUco et MediaPipe sont alignées à partir des horodatages associés aux images de la vidéo. Cette synchronisation permet de relier, pour un même instant, la pose d'un cube, les points humains détectés et la consigne de manipulation.

Les trajectoires reconstruites à partir des marqueurs peuvent contenir du bruit, des pertes courtes de détection ou des sauts ponctuels entre deux images. Pour réduire ces irrégularités, un filtre de Kalman est appliqué aux poses reconstruites [20]. Ce type de filtre combine une prédiction de l'état suivant avec une correction apportée par la nouvelle mesure disponible [21].

Dans ce traitement, la prédiction s'appuie sur l'état précédent du cube, tandis que la correction utilise la mesure fournie par la détection ArUco. Cette étape améliore la continuité des trajectoires et produit des données plus stables pour l'analyse, l'export CSV et la visualisation sous RViz.

4.6 Génération des sorties structurées

La chaîne offline génère plusieurs types de fichiers. Ces sorties sont organisées afin de permettre l'analyse hors ligne, le rejeu dans ROS 2 et la validation visuelle. Le Tableau 4.2 résume les principales sorties produites.

TABLE 4.2 – Principales sorties du traitement offline.

Sortie	**Contenu**
`aruco_poses.csv`	Poses des marqueurs détectés dans les images de la vidéo.
`cube_poses.csv`	Poses reconstruites des cubes dans le repère de travail.
`cube_trajectories.csv`	Trajectoires temporelles des cubes après reconstruction.
`mediapipe_landmarks_raw.csv`	Points humains extraits directement par MediaPipe.
`human_trajectories.csv`	Données temporelles associées au mouvement humain.
`learning_timeseries.csv`	Données synchronisées destinées aux analyses ou aux traitements d'apprentissage.
`quality_report.json`	Indicateurs de qualité du traitement, pertes de détection et informations de contrôle.
`rosbag/`	Fichiers permettant de rejouer les données reconstruites dans un environnement ROS 2 [22].

Ces fichiers gardent le lien entre la manipulation enregistrée, la consigne associée et les données reconstruites. Les fichiers CSV facilitent l'analyse directe, tandis que le rosbag permet de rejouer les résultats avec les outils ROS 2. Le rapport de qualité sert à repérer les manipulations présentant des pertes de détection ou des incohérences visibles.

4.7 Analyse des sorties CSV et validation sous RViz

Avant la visualisation sous RViz, une première validation est réalisée à partir des fichiers CSV générés par le traitement offline. Cette étape permet de vérifier que la manipulation enregistrée produit des données cohérentes : présence des points humains extraits par MediaPipe, détection du cube manipulé, continuité temporelle, fréquence de la vidéo et cohérence entre la consigne demandée et le résultat observé.

Les indicateurs sont calculés sur une manipulation représentative du dataset. Le taux de détection est obtenu en rapportant le nombre d'images contenant l'information attendue au nombre total d'images de la manipulation. Pour MediaPipe, la vérification porte sur les points humains retenus pour le transfert humain–robot. Pour les cubes, elle porte sur la présence d'une pose reconstruite dans les fichiers issus de la détection ArUco.

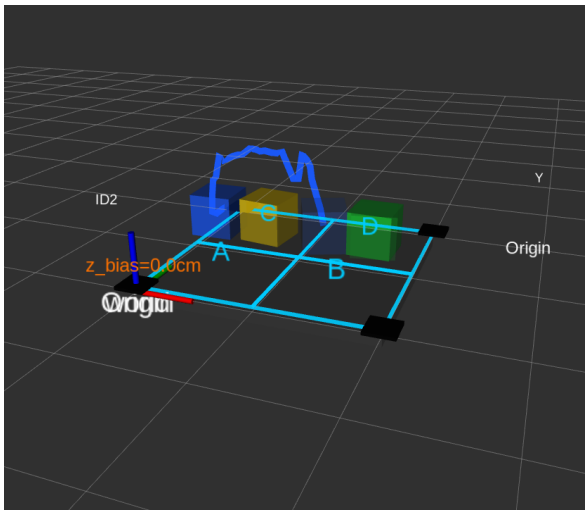
Le Tableau 4.3 présente une synthèse de cette vérification sur une manipulation donnée.

TABLE 4.3 – Validation d'une manipulation à partir des fichiers CSV générés.

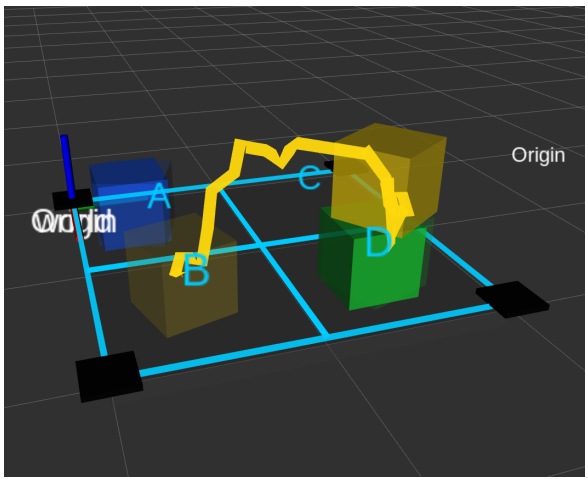
Critère vérifié	**Méthode de vérification**	**Résultat obtenu**	**État**
Fréquence vidéo	- Calcul de la fréquence moyenne à partir des horodatages de la manipulation.	*30 fps mesurés*	✓
Détection du mouvement humain	- Pourcentage d'images où les points humains retenus sont détectés par MediaPipe.	*98.8%*	✓
Détection du cube manipulé	- Pourcentage d'images où une pose du cube est reconstruite à partir des marqueurs ArUco.	*96.4%*	✓
Continuité de la trajectoire	- Vérification des pertes de détection, des sauts ponctuels et des discontinuités après filtrage.	Trajectoire exploitable	✓
Cohérence avec la consigne	- Comparaison entre la consigne demandée et la position finale reconstruite du cube.	Cube arrivé dans la zone cible	✓
Sorties générées	- Présence des fichiers CSV, du rosbag reconstruit et du rapport de qualité.	Fichiers présents	✓

Cette analyse fournit une validation numérique des fichiers produits par le traitement offline. Elle est complétée par une validation visuelle sous RViz, qui permet de vérifier la cohérence spatiale des poses et des trajectoires reconstruites dans le repère de travail.

La Figure 4.6 présente un exemple de visualisation des poses et des trajectoires reconstruites sous RViz.



(a) Poses reconstruites dans le repère de travail.



(b) Trajectoire reconstruite d'une manipulation.

FIGURE 4.6 – Visualisation sous RViz des poses et des trajectoires reconstruites.

La visualisation sous RViz permet de contrôler la cohérence géométrique des résultats. Les poses reconstruites doivent rester compatibles avec l'organisation de la table, les zones de dépôt et le déplacement attendu du cube. Les trajectoires affichées permettent également d'identifier rapidement une erreur de repère, une perte de détection ou une discontinuité temporelle.

4.8 Préparation de la reproduction des trajectoires avec le Niryo Ned2 et collecte des données robot

Après la reconstruction des trajectoires des cubes, une étape complémentaire consiste à les exploiter dans un environnement robotique avec le modèle virtuel du Niryo Ned2 sous RViz. Cette étape permet de préparer la reproduction des mouvements observés et de générer les données robotiques associées.

Le principe consiste à utiliser la trajectoire reconstruite du cube comme référence. À partir des poses extraites par la détection ArUco, on obtient une trajectoire cartésienne décrivant le déplacement du cube dans le repère de travail. Cette trajectoire est ensuite utilisée comme entrée pour le Niryo Ned2 afin de produire un mouvement équivalent dans l'espace de travail du robot.

Les poses cartésiennes peuvent être converties en configurations articulaires à l'aide de MoveIt et de la résolution de cinématique inverse. Chaque point de la trajectoire peut ainsi être associé à une configuration possible du robot. La trajectoire issue de la manipulation humaine est donc transformée en trajectoire robotique, exprimée à la fois par des poses de l'effecteur et par des positions articulaires.

La Figure 4.7 montre le modèle du Niryo Ned2 dans RViz avec la table de manipulation, les zones de travail, les cubes et la trajectoire reconstruite. Cette visualisation permet de suivre la reproduction de la trajectoire dans l'environnement robotique et de préparer l'enregistrement des données générées.

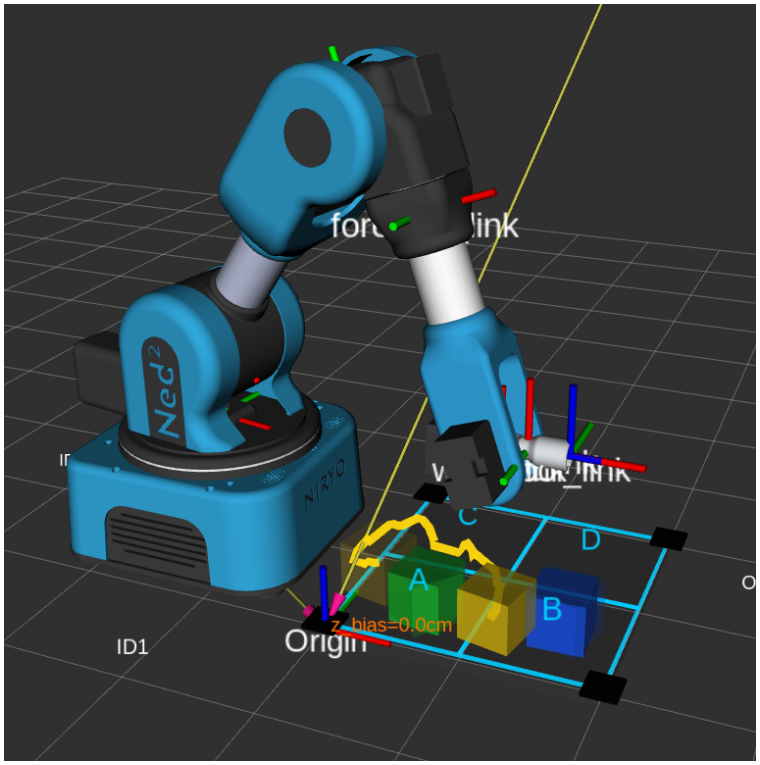


FIGURE 4.7 – Visualisation sous RViz du modèle Niryo Ned2 avec la trajectoire reconstruite du cube.

Pendant cette étape, les données produites par le robot sont enregistrées. Elles comprennent notamment les positions articulaires du Niryo Ned2, leur évolution temporelle et les poses atteintes par l'effecteur terminal pendant la reproduction de la trajectoire. Ces informations permettent de construire une représentation robotique du mouvement observé.

Deux familles de données sont ainsi obtenues. La première correspond au mouvement humain extrait avec MediaPipe, notamment les positions du bras droit, de la main droite et des points caractéristiques sélectionnés. La seconde correspond aux données robotiques générées à partir des trajectoires des cubes détectées par ArUco, comme les positions articulaires du Niryo Ned2 et les poses de son effecteur. Ces deux ensembles sont sauvegardés dans la base de données afin d'être utilisés ensuite par le node de transfert.

Cette étape permet donc de passer d'une trajectoire observée sur les cubes à une trajectoire exploitable par un robot manipulateur. Elle constitue une étape de préparation des données pour le transfert entre le geste humain et le mouvement robotique.

Conclusion personnelle

Ce TER m'a permis de travailler sur un projet complet, allant de la mise en service d'un robot réel jusqu'à la collecte et au traitement de données expérimentales. J'ai pu appliquer ROS 2 dans un contexte concret, avec la configuration de la cellule UR3e, l'intégration de la pince, de la caméra et la mise en place d'une interface opérateur.

Ce projet m'a aussi appris à mieux comprendre les contraintes d'un système robotique réel. La communication réseau, la sécurité pendant les essais, le lancement des différents composants et la validation progressive des mouvements sont des points que j'ai appris à gérer de manière plus pratique.

La partie collecte humain–cube m'a permis de découvrir l'importance de l'organisation des données. J'ai appris à structurer un protocole expérimental, à gérer des sessions, à séparer automatiquement les manipulations et à produire des fichiers exploitables pour l'analyse.

Enfin, les traitements offline m'ont permis de progresser en vision par ordinateur avec ArUco et MediaPipe, ainsi qu'en visualisation avec RViz. Ce projet m'a donc apporté des compétences en robotique, en ROS 2, en traitement de données et en rédaction technique.

Conclusion générale

Ce TER a permis de construire une chaîne expérimentale composée de deux contributions principales. La première concerne la remise en service de la cellule UR3e sous ROS 2. Le robot a été intégré dans un environnement ROS 2 cohérent, avec la mise en place du driver, des contrôleurs, de la pince Robotiq Hand-E, du flux caméra et d'une interface opérateur. Le lancement global de la cellule et la documentation technique associée facilitent désormais la reprise du système par de futurs utilisateurs.

La seconde contribution concerne la mise en place d'une plateforme de collecte humain–cube. Cette plateforme permet de guider un opérateur à travers des consignes de manipulation, d'enregistrer les séquences vidéo, de segmenter automatiquement les manipulations par le marqueur ID99 et d'organiser les données dans un dataset structuré. L'interface web proposée rend le protocole plus simple à utiliser et permet de répéter les sessions dans des conditions homogènes.

Le traitement offline complète cette plateforme en transformant les vidéos brutes en données exploitables. Les marqueurs ArUco sont utilisés pour reconstruire les poses des cubes, MediaPipe permet d'extraire des points caractéristiques du mouvement humain, puis les données sont synchronisées, filtrées et exportées sous forme de fichiers CSV, de rosbags et de résultats visualisables sous RViz. Cette chaîne permet de passer d'une observation expérimentale à des données structurées pouvant être analysées ou reprises dans des travaux ultérieurs.

Le travail réalisé fournit donc une base technique et expérimentale pour la suite du projet. La cellule UR3e peut servir de support robotique sous ROS 2, tandis que la plateforme humain–cube permet de produire des données organisées pour l'étude du transfert humain–robot. Les perspectives principales concernent l'enrichissement du dataset avec davantage de participants et de manipulations, l'amélioration des traitements de reconstruction, puis l'exploitation des trajectoires produites dans des travaux de transfert ou d'apprentissage.

Bibliographie

[1] S. Lengagne, *Mise en service de robots sous ROS 2 et mise en place d'une expérimentation d'observation humaine*, sujet de TER, Institut Pascal, 2026.

[2] G. de Souza Marcos, *Creation of a database of human movements for transfer to robot manipulators*, rapport de stage, Institut Pascal, 2025.

[3] Universal Robots, *UR3e e-Series Datasheet*, fiche technique officielle. En ligne : https://www.universal-robots.com/media/1807464/ur3e_e-series_datasheets_web.pdf, consulté le 03/05/2026.

[4] Open Robotics, *ROS 2 Humble Documentation*. En ligne : https://docs.ros.org/en/humble/, consulté le 05/05/2026.

[5] Universal Robots, *Universal Robots ROS 2 Driver Documentation*. En ligne : https://docs.universal-robots.com/Universal_Robots_ROS2_Documentation/, consulté le 07/05/2026.

[6] Universal Robots, *Universal_Robots_ROS2_Driver*, dépôt GitHub officiel. En ligne : https://github.com/UniversalRobots/Universal_Robots_ROS2_Driver, consulté le 09/05/2026.

[7] ros-controls, *ros2_control documentation*. En ligne : https://control.ros.org/humble/index.html, consulté le 11/05/2026.

[8] IFRA-Cranfield, *ros2_RobotiqGripper*, dépôt GitHub. En ligne : https://github.com/IFRA-Cranfield/ros2_RobotiqGripper, consulté le 15/05/2026.

[9] Robotiq, *Hand-E Instruction Manual*. En ligne : https://assets.robotiq.com/website-assets/support_documents/document/Hand-E_Manual_Generic_PDF_20220111.pdf, consulté le 16/05/2026.

[10] Robotiq, *Retrieving Robotiq Wrist Camera Pictures on a Computer*. En ligne : https://blog.robotiq.com/knowledge/retrieving-robotiq-wrist-camera-pictures-on-a-computer-5-1736280755515, consulté le 20/05/2026.

[11] B. D. Argall, S. Chernova, M. Veloso et B. Browning, *A survey of robot learning from demonstration*, Robotics and Autonomous Systems, vol. 57, no. 5, pp. 469–483, 2009, consulté le 27/05/2026.

[12] M. Mounsif, S. Lengagne, B. Thuilot et L. Adouane, *Universal Notice Networks : Transferring Learned Skills Through a Broad Panel of Applications*, Journal of Intelligent & Robotic Systems, vol. 107, article 18, 2023, consulté le 29/05/2026.

[13] S. Beaussant, S. Lengagne, B. Thuilot et O. Stasse, *Towards zero-shot cross-agent transfer learning via latent-space universal notice network*, Robotics and Autonomous Systems, vol. 184, article 104862, 2025, consulté le 30/05/2026.

[14] A. Lakshmipathy, D. Bauer, C. Bauer et N. S. Pollard, *Contact Transfer : A Direct, User-Driven Method for Human to Robot Transfer of Grasps and Manipulations*, IEEE International Conference on Robotics and Automation, 2022, consulté le 02/06/2026.

[15] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas et M. J. Marín-Jiménez, *Automatic generation and detection of highly reliable fiducial markers under occlusion*, Pattern Recognition, vol. 47, no. 6, pp. 2280–2292, 2014, consulté le 04/06/2026.

[16] OpenCV Contributors, *Detection of ArUco Markers*, documentation officielle OpenCV. En ligne : https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html, consulté le 05/06/2026.

[17] C. Lugaresi et al., *MediaPipe : A Framework for Building Perception Pipelines*, arXiv :1906.08172, 2019, consulté le 08/06/2026.

[18] F. Zhang et al., *MediaPipe Hands : On-device Real-time Hand Tracking*, arXiv :2006.10214, 2020, consulté le 09/06/2026.

[19] V. Bazarevsky et al., *BlazePose : On-device Real-time Body Pose Tracking*, arXiv :2006.10204, 2020, consulté le 10/06/2026.

[20] R. E. Kalman, *A New Approach to Linear Filtering and Prediction Problems*, Journal of Basic Engineering, vol. 82, no. 1, pp. 35–45, 1960, consulté le 18/06/2026.

[21] H. C. Kam, Y. K. Yu et K. H. Wong, *An Improvement on ArUco Marker for Pose Tracking Using Kalman Filter*, IEEE/ACIS SNPD, pp. 65–69, 2018, consulté le 18/06/2026.

[22] Open Robotics, *Recording and playing back data*, documentation officielle ROS 2. En ligne : https://docs.ros.org/en/humble/Tutorials/Beginner-CLI-Tools/Recording-And-Playing-Back-Data/Recording-And-Playing-Back-Data.html, consulté le 24/06/2026.