Évitement d'obstacles par répulsion pour un robot collaboratif à 3 DDL

Corentin Boucher
Département de génie mécanique
École de technologie supérieure
Montréal, Canada

David Olivier
Département de génie électrique
École de technologie supérieure
Montréal, Canada

***Abstract*—Cet article présente une étude de l'utilisation des robots collaboratifs dans un contexte d'assemblage d'un produit quelconque en industrie. Comme la productivité est une priorité dans une ligne d'assemblage, deux avenues sont étudiées dans le but d'améliorer la collaboration humain-robot. D'abord, la perception de mouvements intuitifs du robot par le travailleur humain permet d'augmenter la confiance du travailleur envers son robot collaboratif ce qui améliore la collaboration et le travail effectué. Ensuite, un algorithme d'évitement d'obstacles dynamiques par l'utilisation d'un vecteur de répulsion permet d'augmenter la sécurité de l'humain. Le calcul d'un vecteur de répulsion entre l'obstacle et le point le plus près du robot permet de calculer de nouvelles vitesses aux joints du robot permettant de contourner les obstacles se trouvant dans la trajectoire du robot. Cet algorithme fonctionne indépendemment du planificateur de trajectoires.**

***Index Terms*—Robots collaboratifs, obstacles, vecteur de répulsion.**

I. Introduction

À la recherche de la productivité, les différentes industries optent de plus en plus pour l'implémentation de robots pour effectuer des tâches normalement réalisées par des travailleurs humains. Cependant, certaines tâches requièrent une adaptabilité et une dextérité que l'on retrouve seulement chez l'humain, ce qui rend les robots industriels inaptes à faire ces tâches. C'est dans ces situations, dans le but d'augmenter la productivité du travailleur humain, que les robots collaboratifs peuvent être utilisés.

Cependant, le développement de robots collaboratifs est différent du développement de robots industriels classiques par le soucis du travailleur. Les robots collaboratifs doivent entre autres être sécuritaire pour les humains travaillant dans son enveloppe. Une réduction de force et de vitesse est donc requise pour les robots collaboratifs, tout en ajoutant une intelligence permettant une collaboration plus efficace.

Même si les robots sont conçu pour améliorer la productivité dans l'industrie, ils peuvent changer considérablement l'environnement de travail ce qui pourrait affecter négativement les employés. D'abord, la peur de perdre son emploi en raison de l'automatisation augmente [1] ce qui peut changer l'état d'âme des travailleurs. De plus, certains employés ne veulent pas travailler avec des robots en raison de plusieurs facteurs notamment une plus haute confiance envers les humains et le manque de fluidité du travail d'équipe avec un robot [1]. Donc, une meilleure interaction avec le robot permettrait d'améliorer cet aspect et donc d'améliorer la collaboration humain-robot.

Il y a trois défis principaux dans le développement des robots collaboratifs, soit la sécurité des travailleurs, la confiance de l'humain envers l'automatisation et la productivité de la collaboration [2]. Plusieurs éléments sont à prendre en compte pour répondre à ces trois défis. Dans ce papier, nous présentons une revue de littérature sur les concepts clés de l'interaction humain-robot, ainsi qu'un algorithme d'évitement d'obstacles pour un robot collaboratif 3DOF. On note que nous allons également présenter les simulations que nous avons effectuer afin de valider notre algorithme.

A. Interaction humain-robot (HRI)

La collaboration humain-robot (HRC) demande un certain niveau d'interaction et donc de communication entre le travailleur humain et le robot manipulateur. Comme mentionné dans [2], les défis des robots collaboratifs sont la sécurité, la confiance envers le robot et la productivité. L'interaction humain-robot influence toutes ces caractéristiques en essayant de créer un environnement de travail intuitif pour l'humain. Un élément important de l'interaction est la communication entre les différents partis, ici l'humain et le robot.

Il y a deux sections à la communication, soit l'humain vers le robot, par l'envoie de commande ou la réponse intuitive du robot à partir des actions de l'humain, et le robot vers l'humain (Robot to Human ou RtH). Cet élément de la communication est souvent négligé, mais il est tout aussi important que la commande du robot.

D'abord, la communication des états du robot et la reconnaissance des commandes de l'utilisateur sont requises pour une collaboration sécuritaire et efficace. De plus, une communication bien conçue permet une interaction plus engagée ce qui augmente la confiance de l'utilisateur envers le robot collaboratif [3]. Donc, en augmentant la confiance et la sécurité de l'utilisateur, l'hypothèse est que la productivité sera aussi améliorée, permettant ainsi de toucher aux trois défis des robots collaboratifs.

Pour concevoir cette communication, l'interprétation des mouvements du robot par l'humain est un élément crucial.

Un travail pionnier de l'interaction humain-robot indique que des principes d'animation vidéos permettent de créer des mouvements plus intuitif donnant vie aux robots [4]. De plus, [3] utilise des mouvements inspirés des animaux dans le but de créer une interaction intuitive avec l'utilisateur. Dans ce travail, le regard du robot est utilisé dans le but de diriger l'utilisateur vers un certain endroit de manière intuitive.

Un autre aspect crucial de l'interaction humain-robot lors d'une collaboration est la gestion des erreurs. D'abord, les différentes erreurs du robot vont pousser l'humain vers différentes réactions. Plus l'erreur est importante, plus le travailleur aura une réponse rapide et qui escaladera avec la situation [5]. Le robot doit donc éviter les erreurs sévères pour empêcher l'humain de répondre trop brusquement.

De plus, la gestion de l'erreur humaine est un aspect très délicat lors de la collaboration. Premièrement, si l'humain croit que le robot a fait une erreur, il va s'attendre à ce que celui-ci change son comportement lors du prochain essai. Cependant, si le robot croit que l'humain a effectué l'erreur, il doit décider s'il changera son comportement ou non. Ceci fait en sorte que le robot doit communiquer avec l'utilisateur en cas d'erreur. Cependant, si le robot attribue l'erreur à l'humain, celui-ci sera moins enclin à vouloir travailler avec le robot. Alors, un robot qui s'attribue l'erreur mets le travailleur plus en confiance [6] et donc améliorera la collaboration.

II. Contexte

Les robots collaboratifs peuvent être utilisés pour plusieurs applications, comme l'assistance de personnes à mobilité réduite ou l'utilisation de plateformes mobiles munies d'un manipulateur. Cependant, l'utilisation la plus courante pour ces robots est dans les lignes d'assemblages où plusieurs travailleurs sont assistés par des bras robotiques afin d'améliorer la productivité. Dans un tel contexte, la prévention de collisions machine-humain est un cas d'utilisation idéal pour les algorithmes d'évitement d'obstacles comme celui que nous proposons plus loin.

Sur le plan pratique, nous avons choisi d'implémenter notre algorithme d'évitement d'obstacles sur un robot planaire à trois degrés de liberté. La simplicité de ce robot nous a permit de mettre l'emphase sur le développement et l'implantation de l'algorithme plutôt que sur la cinématique et la dynamique.

A. Cinématique du robot

La figure 1 montre les détails géométriques du robot planaire que nous avons choisi pour la simulation de l'évitement d'obstacles. Chaque segment a une longueur de 1 m dans le but de simplifier les matrices dans les différents calculs. Pour réaliser la simulation, il faut d'abord effectuer la cinématique inverse et donc passer par la cinématique directe. La matrice jacobienne des vitesses linéaires est aussi déterminée.

À l'aide du tableau 1, la cinématique directe est obtenue à l'aide de la convention Denavit-Hartenberg modifiée :

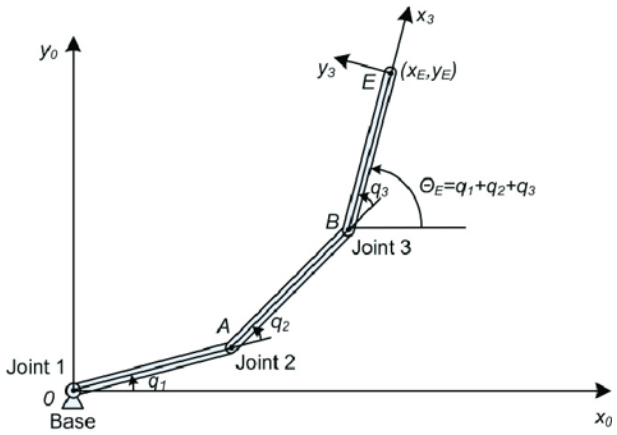

Fig. 1. Robot planaire à 3 DDLs [7]

TABLE I
Paramètres de la convention Denavit-Hartenberg modifiée

Lien	a	α	d	θ
1	0	0	0	$\theta 1$
2	1	0	0	$\theta 2$
3	1	0	0	$\theta 3$
4	1	0	0	0

$$T_4^0 = \begin{bmatrix} c_{123} & -s_{123} & 0 & c_{123}+c_{12}+c_1 \\ s_{123} & c_{123} & 0 & s_{123}+s_{12}+s_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

Ensuite, en effectuant les dérivées partielles des positions en x et y que l'on retrouve dans (1), la jacobienne des vitesses linéaires est déterminées :

$$J^0 = \begin{bmatrix} -s_{123}-s_{12}-s_1 & -s_{123}-s_{12} & -s_{123} \\ c_{123}+c_{12}+c_1 & c_{123}+c_{12} & c_{123} \end{bmatrix} \quad (2)$$

Pour effectuer les mouvements et déplacer le robot selon la position voulue, la cinématique inverse doit être déterminée. Comme le robot a trois degrés de libertés, il faut trois informations pour pouvoir déterminer l'angle de chacun des joints soit x, y et θ_e. D'abord, l'orientation de la pince du robot (θ_e) est déterminée par :

$$\theta_e = atan2(s_{123}, c_{123}) \quad (3)$$

Ce qui permet ensuite de déterminer la position du troisième joint :

$$x_3 = x - cos(\theta_e) \quad (4)$$

$$y_3 = y - sin(\theta_e) \quad (5)$$

À l'aide de ces positions, les angles θ_1 et θ_2 peuvent être calculés suivant la méthode décrites dans [8] avec les équations :

$$\theta_1 = tan^{-1}(\frac{y_3}{x_3}) - cos^{-1}(\frac{x_3^2+y_3^2}{2\sqrt{x_3^2+y_3^2}}) \quad (6)$$

$$\theta_2 = \pi - cos^{-1}(\frac{2 - x_3^2 - y_3^2}{2}) \quad (7)$$

Ce qui permet de finalement trouver le troisième angle à l'aide des angles θ_1, θ_2 et θ_e déjà calculées :

$$\theta_3 = \theta_e - \theta_1 - \theta_2 \quad (8)$$

B. Contrôle du robot

Nous utilisons un contrôle proportionnel discret (équation 9) pour permettre à l'effecteur du robot de se déplacer d'un point de départ (A) jusqu'à un point cible (B). On note que les points (A) et (B) utilisés dans nos simulations sont illustrés dans la figure 2. Dans l'équation 9, les vecteurs θ, θ_{-1}, et θ_{cible} contiennent les positions angulaires des articulations du robot et représentent respectivement : les positions angulaires actuelles, les positions angulaires de la dernière itération et les positions angulaires permettant d'atteindre le point cible (B). On note également que Δt représente l'intervalle de temps entre les itérations du calcul de commande et que Kp est une constante proportionnelle.

$$\theta = \theta_{-1} + Kp \cdot (\theta_{cible} - \theta_{-1}) \cdot \Delta t \quad (9)$$

Où

$$\theta = [\theta_1, \theta_2, \theta_3]^T; \theta_{-1} = [\theta^1_{-1}, \theta^2_{-1}, \theta^3_{-1}]^T; \\ \theta_{cible} = [\theta^1_{cible}, \theta^2_{cible}, \theta^3_{cible}]^T \quad (10)$$

Pour simplifier la notation et la rendre plus compatible avec l'algorithme d'évitement d'obstacles, nous représentons l'algorithme de commande du robot comme le calcul des vitesses angulaires $\dot{\theta}_{cmd}$ à imposer aux articulations afin de se rendre au point (B) (équation 11).

$$\theta = \theta_{-1} + \dot{\theta}_{cmd} \cdot \Delta t \text{ où } \dot{\theta}_{cmd} = Kp \cdot (\theta_{cible} - \theta_{-1}) \quad (11)$$

C. Tâche effectuée par le robot

Dans le cadre de nos simulations, le déplacement de l'effecteur du point de départ (A) jusqu'au point cible (B) représente la tâche que doit effectuer le robot dans le milieu de collaboration humain-machine. Dans le contexte d'une ligne d'assemblage, ceci pourrait correspondre au travail d'un bras robotique qui déplace des caisses d'un endroit à un autre. La prochaine section décrit la représentation d'obstacles physiques dans ce milieu et détaille un algorithme qui permet au robot d'éviter ces obstacles tout en accomplissant sa tâche.

III. Évitement d'obstacles

Notre algorithme d'évitement d'obstacles est inspiré du travail présenté dans [9]. Cet article propose qu'un vecteur de répulsion, affectant les joints du robot, émane de chaque obstacle afin de faire dévier le robot lorsque celui-ci s'approche trop d'un obstacle. On note que l'amplitude d'un vecteur de répulsion reliant un obstacle au robot varie en fonction de la distance entre l'obstacle et le robot ainsi que la vitesse relative du robot par rapport à l'obstacle. Dans tous les cas, les vecteurs de répulsion empêchent le robot d'entrer en collision avec les obstacles dans ses alentours. Nous avons implémenté notre propre version des modèles proposés dans [9] de sorte à obtenir un mécanisme d'évitement d'obstacle fonctionnel dans le contexte décrit précédemment.

A. Vecteur de répulsion

L'équation 12 exprime la notation du vecteur de répulsion généré par un obstacle. Le vecteur de répulsion possède une norme $|V_{rep}|$ et une orientation $\overrightarrow{s}$ sous forme de vecteur unitaire.

$$\overrightarrow{V_{rep}} = |V_{rep}| \cdot \overrightarrow{s} = |v_{rep1} + v_{rep2}| \cdot \overrightarrow{s} \quad (12)$$

On note que $\overrightarrow{s}$ possède la même orientation que la droite reliant le centre de l'obstacle au point le plus proche à l'obstacle sur le robot. Cette droite correspond au segment $C \rightarrow CP$ de la figure 2 où (C) est le centre de l'obstacle et (CP) est le point du robot le plus proche de l'obstacle. On remarque également, pour cet exemple, que (CP) correspond à l'effecteur du robot. En connaissant l'emplacement des points (C) et (CP), on peut calculer $\overrightarrow{s}$ avec l'équation 13 où $\overrightarrow{r_c}$ et $\overrightarrow{r_{cp}}$ sont les vecteurs de positions associés aux points (C) et (CP).

$$\overrightarrow{s} = \frac{\overrightarrow{r_{cp}} - \overrightarrow{r_c}}{|\overrightarrow{r_{cp}} - \overrightarrow{r_c}|} \quad (13)$$

En pratique, l'utilisation du point (CP) dans les calculs implique qu'il faut connaitre la position exacte des obstacles aux alentours du robot. Intuitivement, pour les applications réelles ceci serait l'étape demandant le plus de ressources (calculs et capteurs).

L'équation 12 indique également que la norme du vecteur de répulsion correspond à la somme de deux grandeurs, soit v_{rep1} qui varie en fonction de la distance entre (C) et (CP), et v_{rep2} qui varie en fonction de la vitesse relative de (CP) par rapport à (C). Cependant, avant de pouvoir analyser les équations permettant de calculer ces deux grandeurs, il faut identifier cinq grandeurs clés présentes dans la figure 2.

1) d_{min} correspond à la longueur du segment $C \rightarrow CP$ et représente la plus courte distance entre l'obstacle et le robot.
2) d_{cr} correspond au rayon du premier cercle noir alentour du point (C) et représente le périmètre critique dans lequel aucun point du robot ne peut pénétrer. Autrement dit, d_{min} ne peut pas être plus petit que d_{cr}.
3) d_o correspond au rayon du cercle jaune et représente la distance à partir de laquelle la grandeur v_{rep1} est non nulle. On note que la grandeur de ce rayon varie dans l'intervalle $[d_1, d_2]$ en fonction de la vitesse relative obstacle-robot v_{rel} (équation 14).
4) l_1 correspond également au rayon du cercle jaune (pour cet exemple) et représente la fin de la zone d'amortissement de la grandeur v_{rep2}.
5) l_2 Correspond au rayon du cercle bleu et représente le début de la zone d'amortissement de la grandeur v_{rep2}.

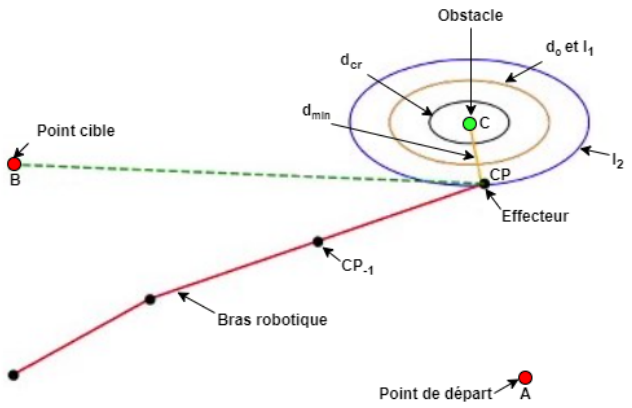

Fig. 2. Extrait annoté de nos simulations illustrant la représentation du bras robotique, des obstacles et des grandeurs clés de notre modèle mathématique. Les valeurs (en mètres) des différents rayons sont : d_{cr}= 0.2, d_1=0.4, d_{max}=12, l_1=d_1 et l_2=$d_1 + 0.2$.

$$d_0 = \begin{cases} d_2, & d_1 - c_v \cdot v_{rel} \geq d_2 \\ d_1 - c_v \cdot v_{rel}, & v_{rel} < 0 \text{ où } c_v = cst \\ d_1, & v_{rel} \geq 0 \end{cases} \tag{14}$$

La grandeur v_{rep1} se calcule via l'équation 15 où la distance d_{min} joue le rôle de variable. On remarque que quand la valeur de d_{min} tend vers d_{cr}, v_{rep1} tend vers ∞. En pratique, ceci garantit que le robot ne puisse jamais s'approcher à une distance inférieure à d_{cr} de l'obstacle, soit $d_{min} < d_{cr}$.

$$v_{rep1} = \begin{cases} k_1 \cdot \left(\frac{d_0}{d_{min} - d_{cr}} - 1\right), d_{min} - d_{cr} < d_2 \\ 0, \quad d_{min} - dcr \geq d_0 \end{cases} \quad k_1 = cst \tag{15}$$

La grandeur v_{rep2} se calcul via l'équation 16 où la vitesse relative obstacle-robot v_{rel} joue le rôle de variable. On note également que, dans l'équation (c) est une grandeur d'amortissement qui dépend de la distance d_{min} (équation 17).

$$v_{rep2} = \begin{cases} -c \cdot k_2 \cdot v_{rel}, v_{rel} < 0 \\ 0, \quad v_{rel} \geq 0 \end{cases} \quad k_2 = cst \tag{16}$$

$$c = \begin{cases} 1, & d_{min} < l_1 \\ \frac{1 + cos\left(\pi \frac{d_{min} - l_1}{l_2 - l_1}\right)}{2}, & l_1 < d_{min} < l_2 \\ 0, & l_2 < d_{min} \end{cases} \tag{17}$$

En général, quand le bras robotique se déplace à basse vitesse, seule la grandeur v_{rep1} contribue à $|V_{rep}|$. À plus haute vitesse, la contribution de v_{rep1} entre en jeu en premier (à partir de d_o) et la contribution de v_{rep2} entre en jeu à partir de l_2.

B. Commande de la répulsion

Une fois calculé, le vecteur de répulsion $\overrightarrow{V_{rep}}$ lié à un obstacle est utilisé pour définir les vitesses angulaires $\dot{\theta}_{rep}$ à imposer aux joints du robot afin que celui-ci puisse l'éviter. Ceci est réalisé en interprétant $\overrightarrow{V_{rep}}$ comme une vitesse linéaire à imposer au joint CP_{-1} (voir figure 2) du robot qui précède directement le point (CP). Les vitesses angulaires engendrées par la vitesse linéaire imposée au joint CP_{-1} sont calculés via l'équation 18 où J_{CP-1} est la jacobienne partielle associée au joint CP_{-1}. Ultimement, chaque obstacle dans les alentours du robot est associé à une matrice $\dot{\theta}_{rep}$ dictant les vitesses angulaires des joints pour l'évitement et celles-ci sont sommés pour produire une seule matrice de vitesses angulaires qui prend en compte tous les obstacles (équation 19).

$$\dot{\theta}_{rep} = J^{-1}_{CP-1} \cdot \overrightarrow{V_{rep}} \tag{18}$$

$$\dot{\theta}_{rep} = \dot{\theta}_{rep1} + \dot{\theta}_{rep2} + ... + \dot{\theta}_{repn}, \quad n = \text{nb d'obstacles} \tag{19}$$

Pour que le robot soit en mesure de se rendre au point cible (B) tout en évitant les obstacles, il est nécessaire que l'algorithme de commande se serve de $\dot{\theta}_{cmd}$ (pour se rendre à la cible (B)) et de $\dot{\theta}_{rep}$ (pour éviter les obstacles). Nous avons donc choisi d'alterner les deux sources de vitesses angulaires en fonction des valeurs de $\dot{\theta}_{rep}$ (équation 20). Plus précisément, quand $\dot{\theta}_{rep} = [0, 0, 0]$ (tous les vecteurs de répulsion sont nuls), l'algorithme fournit $\dot{\theta}_{cmd}$ aux articulations du robot, et $\dot{\theta}_{rep}$ dans le cas contraire.

$$\theta = \theta_{-1} + \dot{\theta} \cdot \Delta t \text{ où}$$
$$\dot{\theta} = \begin{cases} \dot{\theta}_{cmd} = \gamma \cdot Kp \cdot (\theta_{cible} - \theta_{present}), \dot{\theta}_{rep} = [0, 0, 0] \\ \dot{\theta}_{rep}, \quad \dot{\theta}_{rep} \neq [0, 0, 0] \end{cases} \tag{20}$$

Quand on compare les équations 9 et 20, on remarque que nous avons ajouté un facteur γ à la définition $\dot{\theta}_{cmd}$. Celui-ci sert à éviter des changements de vitesse brusques quand l'algorithme de commande transitionne de $\dot{\theta}_{rep}$ à $\dot{\theta}_{cmd}$ en imposant un régime transitoire (équation 21). Dans l'équation, (i) correspond à l'index de l'itération actuelle de l'algorithme de commande et i_{rep} correspond à l'index de l'itération où la transition de $\dot{\theta}_{rep}$ à $\dot{\theta}_{cmd}$ a eu lieu.

$$\gamma = 1 - e^{\frac{i - i_{rep}}{15}} \tag{21}$$

C. Simulation

Pour tester l'algorithme d'évitement d'obstacle, nous l'avons implémenté dans une simulation sur le manipulateur robotique planaire à trois degrés de liberté présenté précédemment. Au niveau technique, pour construire nos simulations, nous nous sommes servis de la librairie Python Robotics [10] qui offre plusieurs exemples de code utiles pour les simulations en robotique. Plus précisément, nous avons adapté une simulation qui permettait de contrôler les mouvements d'un robot planaire via sa cinématique inverse, pour y inclure des obstacles dynamiques et les calculs associés aux vecteurs de répulsion.

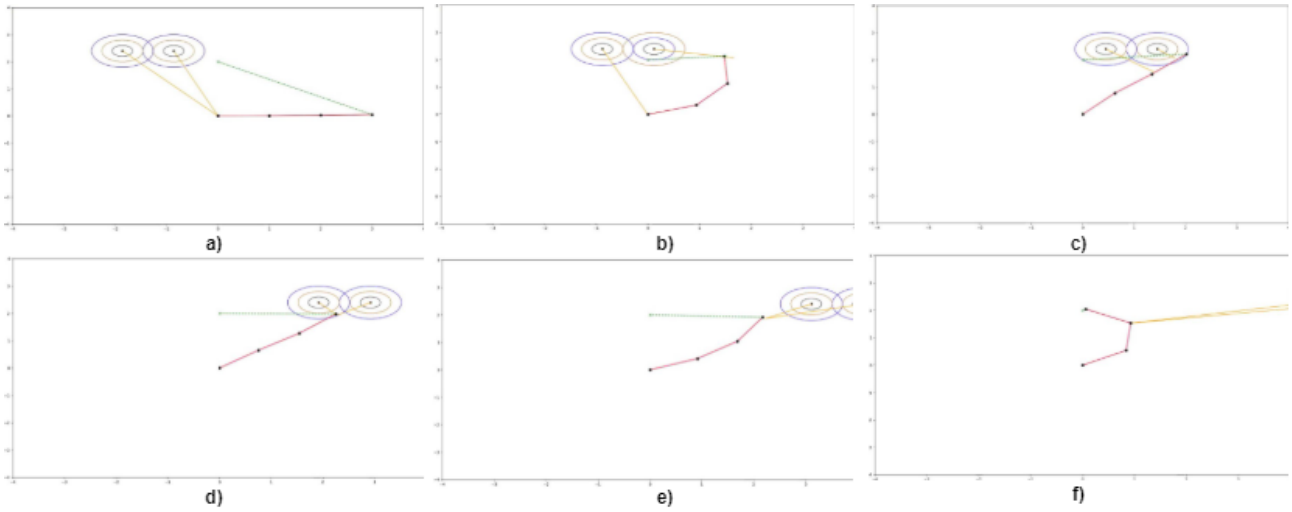

Fig. 3. Extraits séquentiels de l'algorithme d'évitement d'obstacle en action.

La figure 3 illustre notre algorithme d'évitement d'obstacles en action dans une simulation. On note que pour les sous-figures a), b), e) et f) les vitesses des joints du robot correspondent à $\dot{\theta}_{cmd}$ et que pour les sous-figures c) et d) les vitesses correspondent à $\dot{\theta}_{rep}$. On observe donc l'effecteur du robot converger vers le point cible, tout en évitant les deux obstacles mobiles lorsqu'il entre dans la zone d'activation de leurs vecteurs de répulsion.

Les obstacles que nous avons insérés dans la simulation respectent le modèle présenté à la figure 2 et se déplacent à une vitesse constante de 0.7 m/s sur l'axe x. En termes de paramètres de grandeur, leur distance critique d_{cr} est de $0.2m$, et d_o doit être compris dans l'intervalle [d_1 = $0.4m$, d_2 = $0.6m$]. Ceci assure, entre autres, que le robot n'est jamais à moins de $0.2m$ d'un obstacle. De plus, nous avons inséré deux obstacles plutôt qu'un seul afin de tester l'addition de vecteur de répulsion.

Au niveau mathématique, la simulation effectue le calcul des vitesses angulaires à imposer à chaque joint du robot en fonction des propriétés des obstacles dans ses alentours (équation 20). Par la suite, tout comme à l'équation 20, les vitesses angulaires $\dot{\theta}$ obtenues sont multipliées par un incrément de temps discret Δt et le résultat est additionné aux positions angulaires précédentes du robot afin d'obtenir sa nouvelle position. Évidemment, la cinématique du robot est implémentée dans la simulation pour permettre l'utilisation de la jacobienne partielle inverse J^{-1}_{CP-1} requise pour le calcul des vitesses angulaires de répulsions $\dot{\theta}_{rep}$ (équation 18).

Au niveau visuel, la simulation affiche la position des joints et des segments du robot ainsi que la position des obstacles après chaque incrément discret Δt. Ce processus cré une animation fluide qui nous permet d'étudier le comportement de l'algorithme d'évitement d'obstacles.

D. Résultats

À la section (B. Commande de la répulsion), nous avons présenté notre algorithme de contrôle qui, à chaque itération Δt, sélectionne le type de vitesses angulaire parmi $\dot{\theta}_{rep}$ et $\dot{\theta}_{cmd}$ à imposer aux joints du robot (20). On note que lors du développement des modèles mathématiques, nous avons également considéré un algorithme qui calculait simplement la somme des deux types de vitesses angulaires pour ensuite l'imposer aux joints (22). Pour éclairer notre choix d'approche, nous avons donc étudié les mouvements effectués par le robot pour chaque type de contrôle (contrôle par commutation et par addition).

$$\theta = \theta_{-1} + \dot{\theta} \cdot \Delta t \quad \text{où} \quad \dot{\theta} = \dot{\theta}_{cmd} + \dot{\theta}_{rep} \tag{22}$$

La figure 4 illustre les trajets suivis par l'effecteur du robot entre le point de départ (A) et le point cible (B) pour les deux approches de contrôle. Le trajet en rouge a été généré par l'approche par commutation et le trajet en vert a été généré par l'approche par addition. On note que, le trajet en bleu est effectué par l'effecteur quand le robot ne fait aucun évitement d'obstacles.

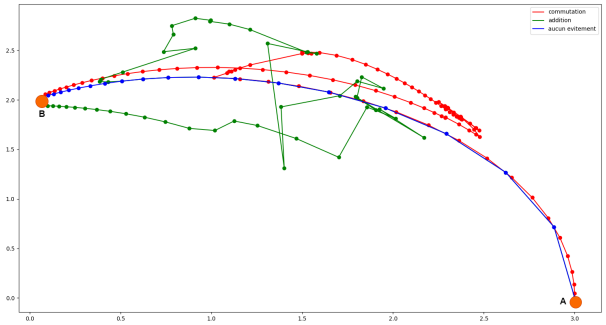

Fig. 4. Trajets effectués par l'effecteur du robot entre le point de départ (A) et le point cible (B), selon le type de contrôle de vitesse, soit : contrôle par commutation (rouge) et contrôle par addition (vert). Le trajet en bleu sert de référence et correspond au trajet sans évitement d'obstacles.

D'abord, on observe que le trajet de l'effecteur pour un contrôle sans évitement d'obstacles (trajet en bleu) ne fait

pas une ligne droite vers son objectif. Cependant, ceci reste le chemin le plus rapide, car les trois joints se déplacent en même temps pour se rendre à leur position finale désirée. Si l'effecteur faisait le chemin le plus court dans le plan XY, ceci ferait en sorte que certains joints devraient être immobiles à certains moments pour suivre la trajectoire. Bref, ce trajet est le plus efficace et permet une bonne prédictibilité du mouvement du robot.

Ensuite, on observe, à travers son trajet (trajet en vert), que le contrôle par addition des vitesses angulaires génère des mouvements erratiques. D'ailleurs, ce genre de mouvement ne serait pas envisageable pour des applications réelles en raison des changements brusques de position qui créent des accélérations trop élevées et qui ajoutent des déplacements non nécessaires, rendant le robot inefficace et dangereux. En effet, le trajet est difficilement prévisible et ne permettrait pas un utilisateur de se sentir en sécurité dans l'enveloppe du robot.

Finalement, on observe, à travers son trajet (trajet rouge), que le contrôle par commutation des vitesses angulaires génère un mouvement sécuritaire sans changements brusques de position. En effet, le tableau 2 indique que le trajet par commutation comporte 15 changements de signes de la vitesse de l'effecteur, ce qui est significativement plus bas que le trajet par addition qui comporte 27 changements de signe. On note que le compte des changements de signe des vitesses linéaires (en X et en Y) de l'effecteur est une façon de quantifier à quel point ses mouvements sont erratiques lors du trajet de (A) à (B). À titre de référence, le trajet sans évitement d'obstacles ne comporte qu'un seul changement de signe dans les vitesses.

TABLE II
NOMBRE DE CHANGEMENTS DE SIGNE DES VITESSES LINÉAIRES (X,Y) DE L'EFFECTEUR SELON LE TYPE DE CONTRÔLE

Type de contrôle	Nombre de changements
Commutation	15
Addition	27
Sans évitement	1

Le trajet généré par la commande par commutation suit de près le trajet de l'effecteur sans évitement d'obstacles, mais effectue les mouvements nécessaires pour ne pas s'approcher sous la distance critique des obstacles. Ce type de trajet permettrait à l'utilisateur de prévoir les mouvements du robot et donc de se sentir en sécurité lors de sa collaboration. Ainsi, il est clair que le contrôle par commutation des vitesses est le plus profitable. Cependant, ce contrôle pourrait être plus efficace en ajoutant un vecteur d'attraction dirigeant l'effecteur vers une trajectoire prédéfinie comme présenté dans [9]. L'attraction permettrait à l'effecteur d'éviter les obstacles sans s'éloigner du but ce qui rendrait le déplacement plus efficace et plus rapide.

IV. CONCLUSION

Ce travail avait pour but de présenter les robots collaboratifs et de trouver des pistes d'amélioration dans le but d'augmenter la productivité d'une collaboration humain-robot dans une ligne d'assemblage. Après avoir étudié la littérature sur l'interaction humain-robot, certaines techniques ressortent comme étant plus efficaces pour augmenter la confiance de l'humain envers son collaborateur robotisé comme l'utilisation de mouvements inspirés d'animations vidéos ou des animaux. Le but d'améliorer l'interaction humain-robot est d'améliorer la perception des mouvements du robot pour rendre la collaboration plus intuitive pour l'humain et donc augmenter sa productivité.

Cependant, dans le but d'augmenter la sécurité du travailleur, un algorithme d'évitement d'obstacles dynamiques permet d'assurer que le robot n'entrera jamais en contact avec l'humain. Un vecteur de répulsion est calculé à partir des positions et vitesses entre le robot et l'humain pour ajuster les vitesses de chacun des joints du robot pour éviter tous les obstacles pouvant se poser dans son chemin. Cet algorithme a été testé en simulation sur un robot planaire à trois degrés de liberté.

Pour les travaux futurs, le même algorithme pourrait être appliqué d'abord en simulation à un robot à six ou sept degrés de liberté permettant ainsi de représenter un vrai robot utilisé dans l'industrie. L'algorithme pourrait ensuite être testé sur le robot en temps réel pour vérifier la vitesse de la réponse. De plus, comme ce travail contrôlait le robot seulement en position, un contrôleur plus avancé doit être implémenté avant l'utilisation sur un robot réel. Le contrôleur devra permettre de contrôler les vitesses et accélérations de chacun des joints permettant d'utiliser l'algorithme sur le robot réel.

REFERENCES

[1] Mariah L. Schrum, Glen Neville, Michael Johnson, Nina Moorman, Rohan Paleja, Karen M. Feigh, and Matthew C. Gombolay. Effects of Social Factors and Team Dynamics on Adoption of Collaborative Robot Autonomy. In *Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '21, pages 149–157, New York, NY, USA, March 2021. Association for Computing Machinery.

[2] Shitij Kumar, Celal Savur, and Ferat Sahin. Survey of Human–Robot Collaboration in Industrial Settings: Awareness, Intelligence, and Compliance. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(1):280–297, January 2021. Conference Name: IEEE Transactions on Systems, Man, and Cybernetics: Systems.

[3] Vanessa Sauer, Axel Sauer, and Alexander Mertens. Zoomorphic Gestures for Communicating Cobot States. *IEEE Robotics and Automation Letters*, 6(2):2179–2185, April 2021. Conference Name: IEEE Robotics and Automation Letters.

[4] Leila Takayama, Doug Dooley, and Wendy Ju. Expressing thought: Improving robot readability with animation principles. In *2011 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 69–76, March 2011. ISSN: 2167-2148.

[5] Maia Stiber and Chien-Ming Huang. Not All Errors Are Created Equal: Exploring Human Responses to Robot Errors with Varying Severity. In *Companion Publication of the 2020 International Conference on Multimodal Interaction*, ICMI '20 Companion, pages 97–101, New York, NY, USA, October 2020. Association for Computing Machinery.

[6] Diede P.M. Van der Hoorn, Anouk Neerincx, and Maartje M.A. de Graaf. "I think you are doing a bad job!": The Effect of Blame Attribution by a Robot in Human-Robot Collaboration. In *Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '21, pages 140–148, New York, NY, USA, March 2021. Association for Computing Machinery.

[7] Adrian-Vasile Duka. Neural Network based Inverse Kinematics Solution for Trajectory Tracking of a Robotic Arm. *Procedia Technology*, 12:20–27, December 2014.

[8] Harry H Asada. *Introduction to Robotics*. MIT, 2004.

[9] Mohammad Safeea, Pedro Neto, and Richard Bearee. On-line collision avoidance for collaborative robot manipulators by adjusting off-line generated paths: An industrial use case. *Robotics and Autonomous Systems*, 119:278–288, September 2019.

[10] Atsushi Sakai, Daniel Ingram, Joseph Dinius, Karan Chawla, Antonin Raffin, and Alexis Paques. PythonRobotics: a Python code collection of robotics algorithms. *arXiv:1808.10703 [cs]*, September 2018. arXiv: 1808.10703.