Aplicação de um manipulador robótico para automatização de uma célula de inspeção de peças automotivas

Lucca Garcia Leão [*,***] **Gustavo Medeiros Freitas** [**]
Luiz Fernando Etrusco Moreira [***] **Antônio Otávio Fernandes** [***]

[*] *Programa de Pós-Graduação em Engenharia Elétrica - Universidade Federal de Minas Gerais - Av. Antônio Carlos 6627, 31270-901, Belo Horizonte, MG, Brasil, luccaleao98@ufmg.br*
[**] *Departamento de Engenharia Elétrica, UFMG, Belo Horizonte, MG, Brasil, gustavomfreitas@ufmg.br*
[***] *Invent Vision, Parque Tecnológico de Belo Horizonte BH-TEC, Belo Horizonte, MG, Brasil, (luizf, otavio)@inventvision.com.br*

Abstract: This article presents an industrial inspection cell that utilizes a six degrees of freedom robotic manipulator to automate the inspection of vehicle parts with complex geometry. An inspection is defined as a sequence of poses that must be visited in sequence, allowing an end-effector mounted camera to capture images of the part from various angles. The two most utilized types of movement by industrial robots are implemented and compared in an inspection task. The first type is kinematic control for linear trajectory tracking in Cartesian space, and the second type uses fifth degree polynomials to generate joint space trajectories. The two movements were tested in a simulation environment that faithfully reproduces the real inspection cell. Experimental results show that the joint space movement results in smoother control signals when compared to the linear movement, although producing unpredictable Cartesian space trajectories.

Resumo: Este artigo apresenta uma célula de inspeção industrial que utiliza um robô manipulador de seis graus de liberdade para a automatização da tarefa de inspeção de peças automotivas de geometria complexa. Uma inspeção é definida como uma sequência de poses que devem ser visitadas em sequência, permitindo que uma câmera montada no efetuador do robô capture imagens de diferentes faces da peça. Os dois tipos de movimento mais utilizados por robôs industriais são implementados e comparados dentro em uma tarefa de inspeção. Uma delas é o controle cinemático para seguimento de trajetórias lineares no espaço Cartesiano, e a outra utiliza polinômios de quinto grau para gerar trajetórias no espaço das juntas. Os dois tipos de movimento foram testados em um ambiente de simulação que reproduz com fidelidade a célula de inspeção em questão. Resultados experimentais mostram que o movimento no espaço das juntas resulta em sinais de controle mais suaves quando comparados ao movimento linear, apesar de seguir uma trajetória imprevisível no espaço Cartesiano.

Keywords: Robotic Manipulator, Process Automation, Quality Control, Kinematic Control, Machine Vision

Palavras-chaves: Robô Manipulador, Automação de Processos, Controle de Qualidade, Controle Cinemático, Visão de Máquina

1. INTRODUÇÃO

Nas últimas décadas, robôs manipuladores têm sido amplamente utilizados na indústria, devido a sua capacidade de realizar tarefas repetitivas com precisão e agilidade, aumentando a eficiência em linhas de montagem. Mais recentemente, com a consolidação de tecnologias da Indústria 4.0 (Lasi et al., 2014), os robôs manipuladores têm sido utilizados também em tarefas de inspeção, aliados a sistemas de visão de máquina. Nesse tipo de aplicação, um sensor (ultrassônico, infravermelho, ou de visão) é utilizado para capturar informações sobre a superfície ou volume do objeto. O robô possui a função de posicionar o sensor para inspecionar diferentes partes do objeto.

A visão de máquina é uma tecnologia que combina conceitos de ótica, visão computacional, e análise de imagens com o objetivo de extrair informações das imagens de

⋆ Este artigo teve o apoio da Universidade Federal de Minas Gerais (UFMG), do Programa de Pós-Graduação em Engenharia Elétrica (PPGEE), da Invent Vision (`http://www.inventvision.com.br/`), da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, do Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brasil (CNPq), e da Fundação de Amparo à Pesquisa do Estado de Minas Gerais - Brasil (FAPEMIG).

forma automatizada. O tipo de informação extraída varia de acordo com o tipo de aplicação, como a detecção de defeitos em peça, análise dimensional, e identificação da posição e orientação de objetos. O setor automotivo foi um dos primeiros a utilizar extensivamente a visão de máquina (Shafi, 2004), e atualmente, é uma das técnicas mais utilizadas na indústria para soluções de inspeção, monitoramento, controle de qualidade e automação de processos devido ao alto nível de flexibilidade e repetibilidade que proporciona, já sendo utilizada com sucesso para controle de qualidade de alimentos e produtos agrícolas (Patel et al., 2012), e para detecção de rachaduras em pontes (Oh et al., 2009).

Este artigo tem como objeto de estudo uma célula de inspeção de peças automotivas, desenvolvida pela empresa Invent Vision, situada no Parque Tecnológico de Belo Horizonte (BH-TEC), um condomínio de empresas e centros de pesquisa, fundado por uma parceria entre a Universidade Federal de Minas Gerais, Governo de Minas Gerais, Prefeitura Municipal de Belo Horizonte, Sebrae e Fiemg.

Dentre os componentes básicos para o funcionamento do sistema de visão em questão, podem ser citados uma fonte de iluminação artificial, um computador com capacidade para processamento de imagens, e uma câmera, que normalmente permanece fixa. Em geral, esse tipo de sistema é satisfatório quando o objeto a ser analisado possui uma geometria simples e toda a região de inspeção pode ser capturada na cena. Por outro lado, quando são inspecionadas peças com geometria mais complexa, com concavidades e regiões de inspeção com diferentes orientações, a utilização de câmeras fixas se torna uma limitação.

Uma possível solução que possibilita a escalabilidade de um sistema de visão de máquina para a inspeção de objetos cada vez mais complexos é a utilização de um robô manipulador, com a câmera montada em seu efetuador, permitindo que seja posicionada em diferentes poses dentro do espaço de trabalho. Esse tipo de montagem já foi implementada com sucesso para a inspeção de objetos usando um *Scanner 3D* (Kuts et al., 2016) e na inspeção de peças de aeronaves utilizando sondas ultrassônicas (Mineo et al., 2015).

Neste artigo, é apresentado o processo de automação de uma tarefa em uma célula de inspeção industrial, utilizando um manipulador robótico para posicionamento da câmera. Para realizar a inspeção, o robô precisa visitar uma série de configurações pré-definidas, e para isso, dois tipos de controle diferentes foram implementados. O primeiro deles é o controle ponto-a-ponto no espaço das juntas, e o segundo é o controle cinemático para seguimento de uma trajetória linear, que são os tipos de movimentos mais comuns utilizados em robôs comerciais. Um ambiente de simulação que reproduz a célula de inspeção com fidelidade foi construído, e os dois tipos de controle foram testados e comparados numa tarefa que demanda diferentes posições e orientações da câmera. Os resultados de simulação mostram que o movimento no espaço das juntas gera comandos mais suaves para o robô, enquanto o movimento no espaço Cartesiano gera comandos mais agressivos, convergindo para uma trajetória conhecida.

Este artigo está dividido em 6 Seções. A Seção 2 apresenta a célula de inspeção, e introduz o conceito de tarefa que deve ser realizada. A Seção 3 apresenta a modelagem matemática do robô utilizado. A Seção 4 introduz as estratégias de controle no espaço Cartesiano e espaço das juntas utilizadas para movimentação do robô. A Seção 5 mostra o ambiente de simulação criado e os resultados obtidos para as estratégias de controle usadas. Por fim, a Seção 6 destaca as conclusões e propõe trabalhos futuros.

Figura 1. Célula de inspeção com robô Kuka KR4 R600.

2. CÉLULA DE INSPEÇÃO AUTOMATIZADA

A célula de inspeção pode ser vista na Figura 1. Nela, foi instalado um robô KR4 R600, da Kuka Robotics, com uma câmera IP Industrial Invent Vision v200 de 5MP, e um conjunto de refletores montado em seu efetuador. A célula possui um sistema de automação de segurança, comandado por um CLP Siemens S7-1200, e um computador com sistema operacional Windows 10 que executa a plataforma de inspeção e controle Inspecta (Figura 2). A comunicação entre o CLP e a plataforma de inspeção é realizada via protocolo Modbus, e o CLP envia comandos para o robô através de uma rede Profinet.

Uma tarefa de inspeção é definida como uma sequência de configurações que deve ser realizada pelo robô. Em cada uma dessas configurações, a câmera deve capturar uma nova imagem da peça, que é enviada para a plataforma de inspeção via protocolo TCP/IP e processada em tempo real. Nessa etapa, algoritmos de visão computacional são utilizados para detectar possíveis problemas de manufatura da peça, como a ausência de parafusos, mau posicionamento de presilhas de encaixe, alterações de cor, e falhas de injeção. Tais problemas podem incorrer em uma falha crítica na peça, como por exemplo, na linha de fragilização de um *airbag*, que garante a ativação correta do dispositivo.

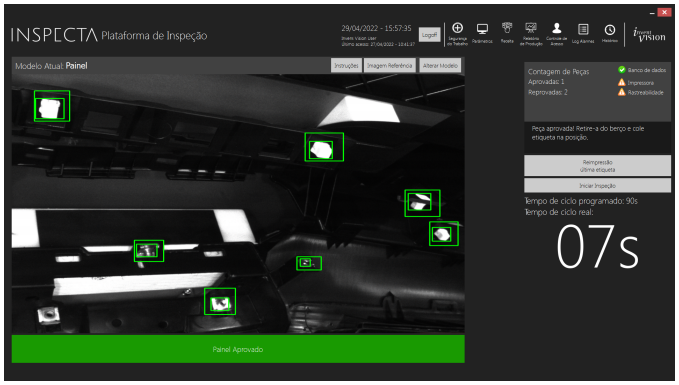

Figura 2. Interface de inspeção.

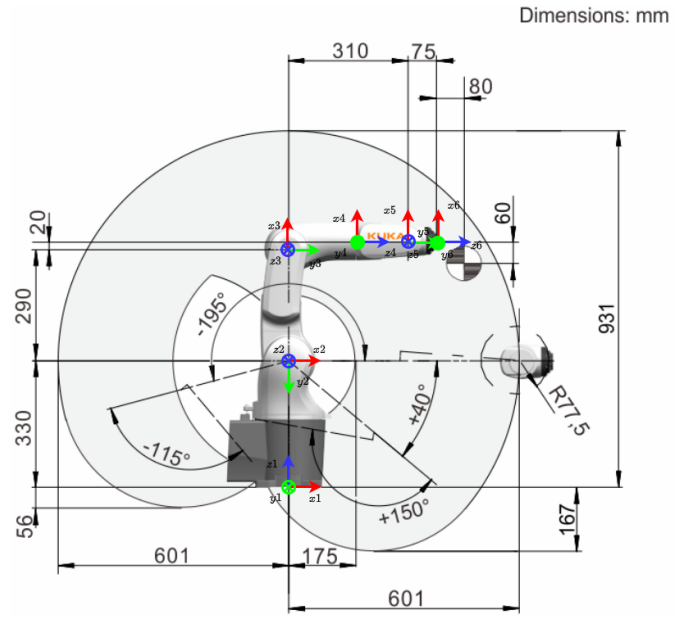

Figura 3. Dimensões e *frames* de referência do KR4 R600. (Adaptado de Kuka KR4 R600 Datasheet, 2021)

3. MODELAGEM DO KR4 R600

Antes de apresentar os controladores implementados, é importante conhecer os parâmetros físicos e cinemáticos que descrevem o robô presente na célula. O KR4 é composto por seis juntas rotativas, conectados por elos de tamanho fixo, como é mostrado na Figura 3.

Uma das formas mais utilizadas de representar as transformações ao longo dos elos é a convenção de Denavit-Hartenberg (Spong et al., 2006). Seguindo essa convenção, a matriz de transformação homogênea entre os *frames* de referência do elo presente em relação ao anterior pode ser obtida pela Equação 1, sendo q_i o ângulo da junta, a_i o tamanho do elo, α_i o ângulo de torção do elo, e d_i o deslocamento do elo. Os parâmetros de Denavit-Hartenberg do KR4 foram extraídos a partir dos dados encontrados em seu *datasheet* (Kuka Robotics, 2021) e podem ser encontrados na Tabela 1. É importante notar que três desses parâmetros, sendo inerentes à montagem do robô, são constantes, e a única variável é o ângulo da junta θ_i.

$$\mathbf{A_i^{i-1}}(q_i) = Rot_z(q_i)Trans_z(d_i)Trans_x(a_i)Rot_x(\alpha_i) \quad (1)$$

Tabela 1. Parâmetros Denavit-Hartenberg do KR4 R600.

Junta	$\theta[rad]$	d[mm]	a[mm]	$\alpha[rad]$
1	$\theta_1 + \pi/2$	330	0	$\pi/2$
2	$\theta_2 - \pi/2$	0	290	0
3	θ_3	0	20	$\pi/2$
4	θ_4	310	0	$-\pi/2$
5	θ_5	0	0	$\pi/2$
6	θ_6	75	0	0

A matriz de transformação homogênea $\mathbf{H}$ define a transformação de coordenadas do efetuador em relação ao *frame* de referência inercial (geralmente fixado na base do robô). Por ser uma matriz composta por um vetor de translação $\mathbf{p_n^0} \in \mathbb{R}^3$ e uma matriz de rotação $\mathbf{R_n^0} \in SO(3)$, $\mathbf{H}$ pertence ao Grupo Especial Euclidiano $SE(3) = \mathbb{R}^3 \times \mathrm{SO}(3)$. Para obter $\mathbf{H}$ basta realizar a multiplicação das matrizes de transformação homogênea dos elos em série:

$$\mathbf{H} = \mathbf{T_n^0} = \mathbf{A_1}(q_1)\dots\mathbf{A_n}(q_n), \quad (2)$$

$$\mathbf{H} = \begin{bmatrix} \mathbf{R_n^0} & \mathbf{p_n^0} \\ 0 & 1 \end{bmatrix}, \quad (3)$$

onde n é o número de graus de liberdade do robô.

A cinemática direta permite obter a pose do efetuador a partir das posições de cada junta do robô. A posição do efetuador em relação à base é dada pelo vetor $\mathbf{p_n^0}$, e a sua orientação é dada pela matriz de rotação $\mathbf{R_n^0}$. É possível decompor $\mathbf{R_n^0}$ em matrizes de rotação elementares e obter os ângulos de *roll*, *pitch* e *yaw* (RPY) do efetuador, e com isso, obter a pose do efetuador:

$$\mathbf{x} = \begin{bmatrix} \mathbf{p_n^0} \\ \boldsymbol{\varphi_n^0} \end{bmatrix}, \quad (4)$$

sendo $\boldsymbol{\varphi_n^0} = \begin{bmatrix} \phi_n^0 & \theta_n^0 & \psi_n^0 \end{bmatrix}^T$ o vetor de orientação do efetuador dado pelos ângulos de RPY.

Definindo $\dot{\mathbf{x}} \in \mathbb{R}^{6\times1}$ como o vetor composto pelas velocidades linear $\mathbf{v_e}$ e angular $\omega_\mathbf{e}$ do efetuador, a equação de cinemática diferencial pode ser utilizada para calcular $\dot{\mathbf{x}}$ através das velocidades das juntas $\dot{\mathbf{q}}$:

$$\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{v_e} \\ \omega_\mathbf{e} \end{bmatrix} = \begin{bmatrix} \mathbf{J_P}(\mathbf{q}) \\ \mathbf{J_O}(\mathbf{q}) \end{bmatrix} \dot{\mathbf{q}}, \quad (5)$$

onde $\mathbf{J_P}(\mathbf{q}) \in \mathbb{R}^{3\times n}$ é a matriz Jacobiana que relaciona a contribuição das velocidades das juntas para a velocidade linear do efetuador, e $\mathbf{J_O}(\mathbf{q}) \in \mathbb{R}^{3\times n}$ é a matriz Jacobiana que relaciona a contribuição das velocidades das juntas para a velocidade angular do efetuador. Com isso, é possível construir a matriz Jacobiana geométrica $\mathbf{J}(\mathbf{q}) \in \mathbb{R}^{6\times n}$:

$$\mathbf{J}(\mathbf{q}) = \begin{bmatrix} \mathbf{J_P} \\ \mathbf{J_O} \end{bmatrix}, \quad (6)$$

que depende da configuração atual do robô.

Para o caso particular de um robô que possui apenas juntas rotativas, e seguindo a convenção de Denavit-Hartenberg, uma coluna i da Jacobiana é calculada por:

$$\begin{bmatrix}\mathbf{J_{P_i}}\\ \mathbf{J_{O_i}}\end{bmatrix} = \begin{bmatrix}\mathbf{z^0_{i-1}} \times (\mathbf{p^0_n} - \mathbf{p^0_{i-1}})\\ \mathbf{z^0_{i-1}}\end{bmatrix}, \tag{7}$$

onde $\mathbf{z^0_{i-1}}$ é o vetor unitário que aponta na direção do eixo $\mathbf{z}$ da junta i, $\mathbf{p^0_{i-1}}$ é o vetor posição da origem do *frame* da junta $i-1$.

4. CONTROLE DE UM ROBÔ MANIPULADOR

Tipicamente, robôs industriais comerciais possuem dois tipos básicos de movimento: ponto-a-ponto seguindo uma trajetória linear no espaço Cartesiano, e ponto-a-ponto no espaço das juntas. Tais movimentos são comumente conhecidos como *"MoveL"* e *"MoveJ"*, respectivamente.

Assumindo que cada junta do robô possui uma malha de controle interna que recebe como referência uma velocidade angular que deve ser atingida pela junta, e computa o torque necessário para garantir o seguimento de referência. Com um ganho alto, essa malha de controle é rápida o suficiente para garantir:

$$\mathbf{u} \approx \dot{\mathbf{q}} \tag{8}$$

Com isso, $\dot{\mathbf{q}}$ pode ser usado diretamente como o sinal manipulado para o controle cinemático do robô.

O erro entre a pose atual do robô pose atual $\mathbf{x}$ e a pose desejada $\mathbf{x_d}$ pode ser definido por:

$$\mathbf{e} = \mathbf{x_d} - \mathbf{x}, \tag{9}$$

e a sua derivada:

$$\dot{\mathbf{e}} = \dot{\mathbf{x}}_\mathbf{d} - \dot{\mathbf{x}}. \tag{10}$$

O vetor erro $\mathbf{e} = [\mathbf{e_P}\ \mathbf{e_O}]^T$ é composto pelo erro de posição e orientação do efetuador. Para o erro de posição, temos a expressão:

$$\mathbf{e_p} = \mathbf{p_d}(\mathbf{t}) - \mathbf{p^0_n}, \tag{11}$$

com derivada:

$$\dot{\mathbf{e}}_\mathbf{p} = \dot{\mathbf{p}}_\mathbf{d}(\mathbf{t}) - \mathbf{v_e}. \tag{12}$$

Por outro lado, a expressão do erro de orientação depende na forma escolhida para representar a orientação do efetuador (ângulos de Euler; ângulo e eixo; ou quatérnio unitário). Para a estratégia de controle implementada neste artigo, foi escolhida a representação por quatérnio unitário, cuja principal vantagem é evitar a ocorrência de representações singulares.

Sendo $\mathbf{R}_d$ e $\mathbf{R}^0_n$ as matrizes de rotação que representam, respectivamente, a orientação desejada e atual do robô, e $Q_d = \{\eta_d, \epsilon_d\}$ e $Q_e = \{\eta_e, \epsilon_e\}$ os quatérnios associados a $\mathbf{R}_d$ e $\mathbf{R}^0_n$. O erro de orientação pode ser obtido pelo quatérnio $\Delta Q = \{\Delta\eta, \Delta\epsilon\}$ que é calculado como:

$$\Delta Q = Q_d * Q_e^{-1}, \tag{13}$$

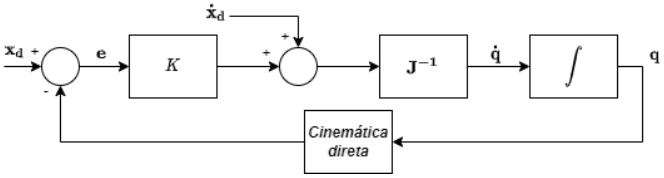

Figura 4. Diagrama de blocos para do controle cinemático.

É possível mostrar que $\Delta Q = \{1, 0\}$ quando $\mathbf{R}_d$ e $\mathbf{R}^0_n$ estiverem alinhadas. Portanto, é suficiente adotar o erro de orientação como $\mathbf{e_O} = \boldsymbol{\Delta}\epsilon$.

4.1 Controle cinemático para seguimento de trajetória

O objetivo de controle é garantir seguimento da trajetória $\mathbf{x} \longrightarrow \mathbf{x_d(t)}$, utilizando $\mathbf{u}$ como entrada para o sistema, seguindo o diagrama mostrado na Figura 4 (Sciavicco et al., 2011).

Substituindo a Equação 5 em 10, obtemos:

$$\dot{\mathbf{e}} = \dot{\mathbf{x}}_\mathbf{d} - \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \tag{14}$$

Assumindo que $\mathbf{J}$ é quadrada e não-singular, a seguinte lei de controle pode ser escolhida:

$$\mathbf{u} = \mathbf{J^{-1}}(\mathbf{q})\bar{\mathbf{u}}, \tag{15}$$

que leva ao sistema linear:

$$\dot{\mathbf{e}} = \dot{\mathbf{x_d}} - \bar{\mathbf{u}}. \tag{16}$$

Escolhendo $\bar{u}$ como:

$$\bar{\mathbf{u}} = \dot{\mathbf{x}}_\mathbf{d} + \mathbf{K}\mathbf{e}, \tag{17}$$

onde $\mathbf{K}$ é o termo proporcional ao erro e $\dot{\mathbf{x}}_\mathbf{d}$ corresponde a uma ação *feedforward*, resultando na lei de controle:

$$\mathbf{u} = \mathbf{J^{-1}}(\mathbf{q})(\dot{\mathbf{x}}_\mathbf{d} + \mathbf{K}\mathbf{e}). \tag{18}$$

A dinâmica do erro em malha fechada pode ser escrita como:

$$\dot{\mathbf{e}} + \mathbf{K}\mathbf{e} = \mathbf{0}, \tag{19}$$

onde $\mathbf{K} = k\mathbf{I_{n\times n}}$. Quando $k > 0$, o sistema é assintoticamente estável:

$$\mathbf{e}(t) = e^{-kt}\mathbf{e}(0), \tag{20}$$

com $\mathbf{e} \longrightarrow \mathbf{0}$ quando $t \longrightarrow \infty$.

Movimentos no espaço Cartesiano estão sujeitos ao risco de singularidades, que ocorrem quando a matriz Jacobiana tem posto menor que n. A presença de singularidades pode impactar negativamente a malha de controle, gerando sinais de controle e torques impraticáveis para as juntas, e impedindo o seguimento correto da trajetória.

4.2 Geração de trajetórias lineares

A lei de controle da Equação 18 é capaz de fazer o efetuador do robô seguir uma trajetória $\mathbf{x_d}(t)$. É possível, portanto, especificar uma curva parametrizada no tempo que será realizada pelo robô, desde que seja contínua e diferenciável.

Em um ambiente industrial, o tempo de execução da tarefa é uma variável importante, pois impacta diretamente no restante da cadeia produtiva. Assim, técnicas que que possibilitam impor restrições de tempo para as trajetórias se tornam mais vantajosas.

O objetivo de uma tarefa é que o robô visite todas as poses alvo em sequência. Sendo assim, a trajetória realizada pelo robô entre os alvos não é crítica para a inspeção. Entretanto, considerando que o robô está inserido em uma célula cercada com paredes e obstáculos internos que limitam em grande parte o seu espaço de trabalho, é de grande valia poder checar a ocorrência de colisões dentro da trajetória entre cada pose. Por esse motivo, o seguimento de trajetórias conhecidas é muito interessante.

Definindo um ponto inicial inicial $\mathbf{p_i}(t_i)$ e posição inicial final $\mathbf{p_f}(t_f)$, sendo $t_f > t_i$, a trajetória retilínea que conecta esses dois pontos atendendo as restrições de tempo inicial e final é:

$$\mathbf{p_d}(t) = \mathbf{p_i} + \frac{(\mathbf{p_f} - \mathbf{p_i})t}{t_f}, \tag{21}$$

e sua derivada:

$$\dot{\mathbf{p}}_\mathbf{d}(t) = \frac{(\mathbf{p_f} - \mathbf{p_i})}{t_f}. \tag{22}$$

A regulação da orientação desejada pode ser feita durante o seguimento da trajetória de posição. Assim, a pose desejada pode ser definida como:

$$\mathbf{x_d}(t) = \begin{bmatrix} \mathbf{p_d}(t) \\ \varphi_\mathbf{d} \end{bmatrix}, \tag{23}$$

e sua derivada:

$$\dot{\mathbf{x}}_\mathbf{d}(t) = \begin{bmatrix} \dot{\mathbf{p}}_\mathbf{d}(t) \\ \mathbf{0} \end{bmatrix}. \tag{24}$$

Essa estratégia permite conhecer a trajetória que será realizada pelo efetuador no espaço Cartesiano, facilitando a checagem de colisões entre o efetuador e obstáculos presentes no espaço de trabalho.

4.3 Movimento ponto-a-ponto no espaço das juntas

O objetivo do movimento ponto-a-ponto no espaço das juntas é levar o robô de uma configuração $\mathbf{q_i}$ até uma configuração desejada $\mathbf{q_d}$, atuando nas juntas de maneira independente. Para gerar a trajetória de cada junta, supomos que no tempo t_0 as variáveis da junta satisfazem:

$$q(t_0) = q_0, \tag{25}$$
$$\dot{q}(t_0) = v_0, \tag{26}$$
$$\ddot{q}(t_0) = \alpha_0, \tag{27}$$

e deseja-se alcançar os valores em t_f:

$$q(t_f) = q_f, \tag{28}$$
$$\dot{q}(t_f) = v_f, \tag{29}$$
$$\ddot{q}(t_f) = \alpha_f. \tag{30}$$

A trajetória deve satisfazer todas as seis restrições impostas nas variáveis das juntas. Para isso, é necessário que ela seja, no mínimo, um polinômio de quinta ordem. Assim, a velocidade e aceleração desejadas são:

$$\dot{q}(t) = a_1 + 2a_2t + 3a_3t^2 + 4a_4t^3 + 5a_5t^4, \tag{31}$$

$$\ddot{q}(t) = 2a_2 + 6a_3t + 12a_4t^2 + 20a_5t^3. \tag{32}$$

E a trajetória da junta i é dada pelo polinômio de quinta ordem:

$$q_i(t) = a_0 + a_1t + a_2t^2 + a_3t^3 + a_4t^4 + a_5t^5, \tag{33}$$

onde $i = 1, \ldots, n$. Os coeficientes $a_0, a_1, a_2, a_3, a_4, a_5$ podem ser determinados a partir dos valores de t_0 e t_f.

5. RESULTADOS EM SIMULAÇÃO

A simulação da célula de inspeção é uma ferramenta que se prova muito útil na etapa de modelagem da tarefa, que consiste na escolha das poses que serão visitadas pelo robô para a captura de imagens. Esse procedimento, em geral, é feito de maneira empírica, posicionando o robô manualmente em diferentes configurações até que a imagem capturada pela câmera seja adequada. Esse processo, entretanto, requer que a célula de inspeção esteja completamente montada, com o robô e a câmera instalados. Com isso, a simulação pode ser utilizada para realizar uma emulação das poses de inspeção de forma independente da célula real.

Para simular a tarefa de inspeção, os dois tipos de movimento foram implementados utilizado o MATLAB, com auxílio do *Robotics Toolbox* (Corke, 1996), e o simulador de robótica CoppeliaSim (Rohmer et al., 2013), ambos com licenças educacionais. O *script* MATLAB se comunica em tempo real com o simulador, enviando comandos e recebendo dados do cenário.

O modelo CAD da célula de inspeção real foi importado dentro do simulador, como visto na Figura 5 e uma peça fictícia que representa um *spoiler* de automóvel foi posicionada em seu interior para ser inspecionada. Por fim, o robô foi fixado na posição correspondente à montagem real, e uma câmera fixada em seu efetuador. Essa configuração permite a prototipagem virtual de células de manufatura e inspeção, por meio da co-simulação entre o MATLAB e o CoppeliaSim, utilizando modelos CAD.

Em seguida, o robô foi colocado em seis configurações com posições e orientações diferentes (Tabela 2), permitindo que a câmera capture imagens de todas as faces da peça.

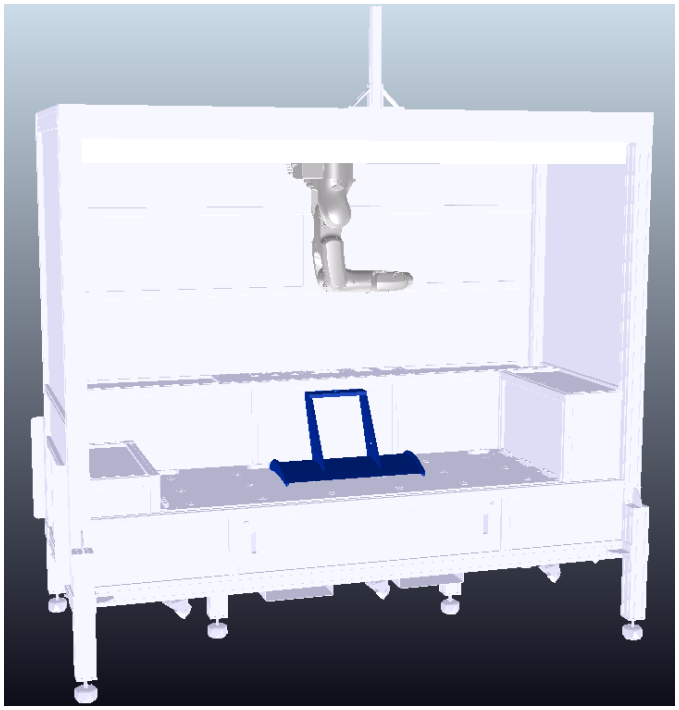

Figura 5. Célula de inspeção no ambiente CoppeliaSim.

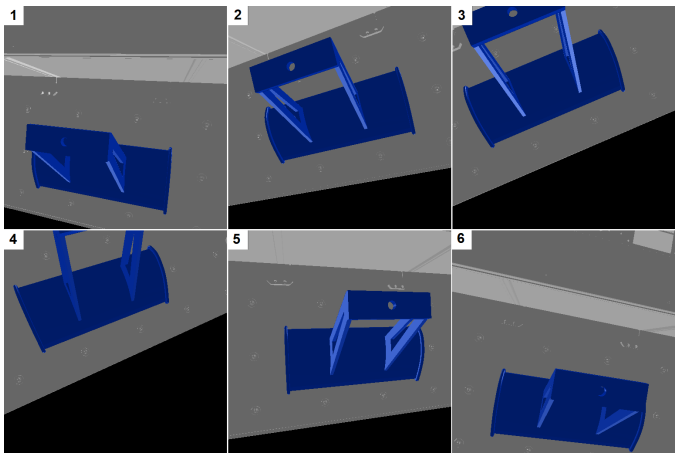

Figura 6. Imagens capturadas na inspeção.

A inspeção foi realizada visitando as poses definidas na Tabela 2, e ao parar em cada pose, um comando para captura de imagem da câmera é enviado ao simulador. A Figura 6 mostra as imagens que foram capturadas durante essa inspeção.

Os erros de posição e orientação, apesar de não serem utilizados no controle no espaço das juntas, foram registrados como métrica de desempenho. A Figura 7 mostra a evolução da pose do efetuador, e a Figura 8 mostra a evolução do erro durante a inspeção. O erro de posição é calculado considerando a distância Euclidiana entre a pose do robô e a pose final desejada. Para as trajetórias definidas pelas Equações 21 e 33 e com os parâmetros da Tabela 2, os dois controladores foram capazes de atingir as poses desejadas. No início do movimento linear, o erro de posição converge exponencialmente até alcançar a trajetória de referência, e então passa a diminuir linearmente até atingir a posição alvo. Na porção final da trajetória, o movimento no espaço das juntas teve convergência mais rápida. Os erros de orientação, por outro lado, tiveram uma convergência mais rápida no movimento linear, enquanto que no espaço das juntas, eles chegam a sofrer um aumento antes de começarem a diminuir.

Tabela 2. Poses para inspeção do *spoiler*.

Pose	x [m]	y [m]	z [m]	ϕ[°]	θ[°]	ψ[°]	t_f [s]
1	0,28	0	0,6	-4,97	-14,86	-89,88	5
2	0,23	0,3	0,6	10,52	-14,31	-85,95	10
3	0	0,4	0,6	19,45	1,32	-86,25	15
4	-0,25	0,4	0,6	18,44	16,28	-86,09	20
5	-0,3	0,2	0,6	-1,75	14,45	-85,90	25
6	-0,2	0	0,6	-16,51	8,10	-83,65	30

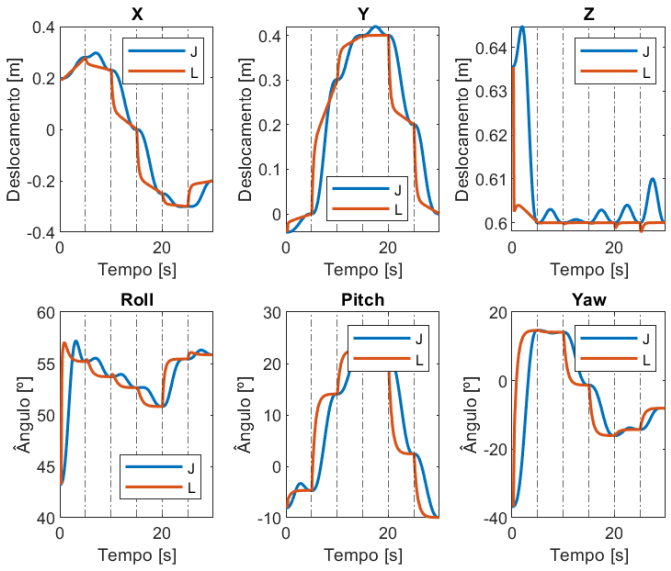

Figura 7. Evolução da pose do efetuador.

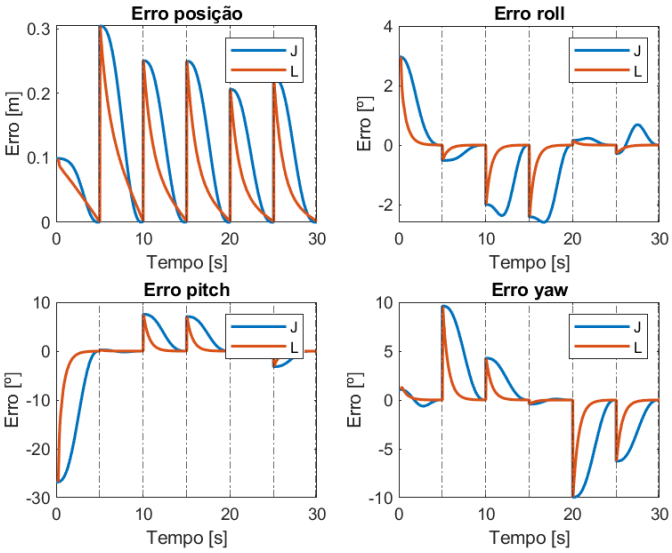

Figura 8. Evolução do erro.

A Figura 9 mostra os sinais de controle que foram aplicados em cada junta durante a inspeção. Dados os valores de ganhos adotados, o movimento no espaço das juntas resultou

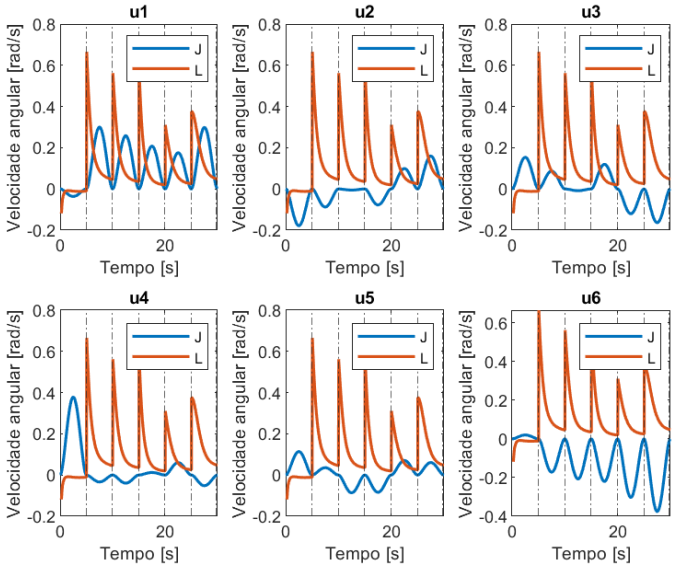

Figura 9. Sinais de controle aplicados nas juntas.

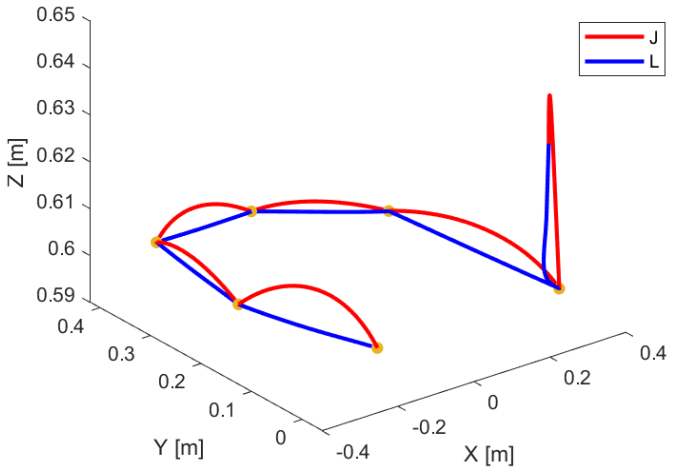

Figura 10. Trajetórias realizadas pelo robô.

em sinais de controle significativamente mais suaves para o robô quando comparados ao movimento linear.

A trajetória realizada pelo robô em cada movimento pode ser visualizada na Figura 10. No espaço das juntas, a trajetória é definida independentemente para cada junta, gerando trajetórias imprevisíveis no espaço Cartesiano.

A Figura 11 evidencia uma situação onde a imprevisibilidade do movimento no espaço das juntas resulta em uma colisão entre a peça e o robô. A trajetória em amarelo mostra o movimento linear no espaço Cartesiano, e a trajetória em vermelho mostra o movimento realizado no espaço das juntas.

6. CONCLUSÃO E TRABALHOS FUTUROS

Este artigo apresentou uma aplicação de um manipulador robótico em uma célula de inspeção automatizada e o desenvolvimento de um ambiente de simulação para a tarefa de inspeção de peças automotivas. O ambiente de simulação utiliza modelos mecânicos fiéis da célula de inspeção e do robô. Foram propostos dois tipos de técnicas de controle de movimento de um manipulador robótico

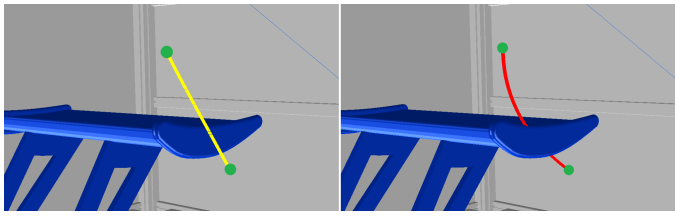

Figura 11. Colisão gerada pelo movimento no espaço das juntas.

amplamente utilizados na indústria: o movimento linear no espaço Cartesiano e o movimento no espaço das juntas.

As duas técnicas foram testadas dentro do ambiente de simulação, realizando uma tarefa de inspeção de uma peça fictícia. Uma câmera instalada no efetuador do robô foi utilizada para capturar imagens da peça durante a inspeção. O robô realizou com sucesso as trajetórias definidas, obedecendo as restrições de tempo impostas para cada trecho e obtendo imagens da peça em diferentes ângulos.

O movimento no espaço Cartesiano se mostrou mais interessante para a aplicação em uma célula de inspeção, devido a previsibilidade na trajetória realizada pelo robô. Uma grande vantagem desse tipo de controle é a capacidade de seguir qualquer tipo de curva paramétrica, desde que seja contínua e diferenciável, e as restrições temporais e físicas do espaço de trabalho sejam fisicamente viáveis para o robô, que está sujeito a problemas relacionados a singularidades. Além disso, pode ser usada para realizar outras tarefas, em que o seguimento de trajetória é crítico, como em um processo de solda arco.

Por outro lado, o movimento no espaço das juntas pode ser utilizado em tarefas mais simples, onde não há grandes preocupações com o caminho que o robô percorre no espaço Cartesiano, desde que chegue na configuração desejada e com sinais de controle mais suaves, como em uma tarefa de *pick and place*.

Em uma futura iteração da célula, o ambiente de simulação poderá ser utilizada para realizar uma emulação da inspeção, facilitando o processo realizado por tentativa e erro com o robô real.

Como trabalhos futuros, diferentes técnicas de controle para seguimento de trajetória serão implementadas, como o Controle Preditivo Baseado em Modelo (MPC) (Poignet and Gautier, 2000), com a finalidade de minimizar o tempo de execução da tarefa. Além disso, serão estudadas técnicas para a estimação de pose da câmera a partir da cena (Lee et al., 2020), que podem compor uma solução de controle baseado em imagem.

REFERÊNCIAS

Corke, P.I. (1996). A robotics toolbox for MATLAB. *IEEE Robotics & Automation Magazine*, 3(1), 24–32.

Kuka Robotics (2021). KUKA KR4 R600 Datasheet. 0000-356-043.

Kuts, V., Tahemaa, T., Otto, T., Sarkans, M., and Lend, H. (2016). Robot manipulator usage for measurement in production areas. *Journal of Machine Engineering*, 16.

Lasi, H., Fettke, P., Kemper, H.G., Feld, T., and Hoffmann, M. (2014). Industry 4.0. *Business & information systems engineering*, 6(4), 239–242.

Lee, T.E., Tremblay, J., To, T., Cheng, J., Mosier, T., Kroemer, O., Fox, D., and Birchfield, S. (2020). Camera-to-robot pose estimation from a single image. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 9426–9432. IEEE.

Mineo, C., Pierce, S., Wright, B., Cooper, I., and Nicholson, P. (2015). PAUT inspection of complex-shaped composite materials through six DOFs robotic manipulators. *Insight-Non-Destructive Testing and Condition Monitoring*, 57(3), 161–166.

Oh, J.K., Jang, G., Oh, S., Lee, J.H., Yi, B.J., Moon, Y.S., Lee, J.S., and Choi, Y. (2009). Bridge inspection robot system with machine vision. *Automation in Construction*, 18(7), 929–941.

Patel, K.K., Kar, A., Jha, S., and Khan, M. (2012). Machine vision system: a tool for quality inspection of food and agricultural products. *Journal of food science and technology*, 49(2), 123–141.

Poignet, P. and Gautier, M. (2000). Nonlinear model predictive control of a robot manipulator. In *6th International workshop on advanced motion control. Proceedings (Cat. No. 00TH8494)*, 401–406. IEEE.

Rohmer, E., Singh, S.P., and Freese, M. (2013). CoppeliaSim (formerly V-REP): a versatile and scalable robot simulation framework. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1321–1326. IEEE.

Sciavicco, L., Siciliano, B., Villani, L., and Oriolo, G. (2011). Robotics: Modelling, planning and Control, ser. Advanced Textbooks in Control and Signal Processing.

Shafi, A. (2004). Machine vision in automotive manufacturing. *Sensor Review*.

Spong, M.W., Hutchinson, S., Vidyasagar, M., et al. (2006). *Robot modeling and control*, volume 3. Wiley New York.