Closed Form HJB Solution for Path Planning of a Robot Manipulator with Warehousing Applications

Prakash, Ravi; Mohanta, Jayant Kumar; Behera, Laxmidhar

DOI
10.1109/CASE49997.2022.9926505

Publication date
2022

Document Version
Final published version

Published in
Proceedings of the IEEE 18th International Conference on Automation Science and Engineering, CASE 2022

2022 IEEE 18th International Conference on Automation
Science and Engineering (CASE)
August 20-24, 2022. Mexico City, Mexico

2022 IEEE 18th International Conference on Automation Science and Engineering (CASE) | 978-1-6654-9042-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/CASE49997.2022.9926505

Closed Form HJB Solution for Path Planning of a Robot Manipulator with Warehousing Applications

Ravi Prakash*, Jayant Kumar Mohanta† and Laxmidhar Behera‡

***Abstract*— Real-time optimal path planning for robotic manipulations in task space is a very fundamental and important problem. In this paper, the problem of generating robot trajectories in an obstacle-ridden environment is formulated under an optimal control framework using Hamilton-Jacobi-Bellman (HJB) equation. The novel contribution of this paper is that a closed form HJB control solution (a necessary and sufficient condition for global optimality of a control solution with respect to a cost function) has been achieved for generating real-time optimal trajectories for a robot manipulator. In contrast with the decoupled end-effector path planning and subsequent trajectory generation, the proposed scheme can exploit sensory input for real-time trajectory generation where the end-effector path as well as the joint trajectory is recomputed online while satisfying the real-time constraints. The stability and the performance of the proposed control framework is shown theoretically via Lyapunov approach and also verified experimentally using a 6 degrees of freedom (DOF) Universal Robot (UR) 10 robot manipulator. It is shown that a significant saving in cost metrics can be obtained over similar trajectory generation approaches from the state-of-the-art with obstacle-ridden environment and also has better performance in high speed tracking applications. Warehouse applications of the proposed scheme in case of static and dynamic targets with respect to the robot manipulator is also included.**

I. Introduction

Motion generation for a task at hand is a very fundamental problem in robotics in order to transfer a robotics system such as robot manipulator from an initial configuration to a desired configuration. The motion includes sequence of robot movements as a function of time namely the *trajectory* which is determined by the task complexities, robot kinematics and desired performance metrics. A very common approach of trajectory generation for a robot manipulator is a decoupled approach [1], [2]. In this approach, the robot end effector path is first determined considering the geometry of the task and the environment e.g. obstacles in the workspace and subsequently trajectory generation in joint space is performed to achieve end-effector path tracking. However, this approach rely heavily on accurate robot models and may give rise to infeasible trajectories. Moreover, this approach is unsuitable for real-time trajectory generation in unstructured and dynamic environments. Almost any real world robotics application in modern day requires sensory measurements of both the internal robot (joints, end-effector) and external sensors (vision) for real-time motion execution in dynamic environments. Therefore, ability to exploit these sensory input in such environments is a desired characteristics for real-time trajectory generation. In other words, the end-effector path as well as the corresponding joint trajectory needs to be recomputed online while satisfying the real-time constraints.

In many real-world robotics applications, the trajectory generation not only needs to guarantee accurate end-effector path tracking but also optimize some pre-defined performnace metrics such as time of execution or energy consumed during the motion. Therefore, trajectory generation methods are often formulated as non-linear optimization problem with a pre-specified cost functional [3]. There are mainly three different families of optimal joint trajectory generation methods: numerical optimization techniques, convex optimization techniques, and optimization techniques based on optimal control theory.

The numerical techniques for trajectory generation for robot manipulator are based on numerical formulations like the Newton-Raphson iteration [4]–[7], predictor-corrector integration [8], [9], gradient based [10], euler and trapezoid method [11], [12]. These solutions are generally not so reliable and has stability and efficiency issues. The failure of such algorithm does not imply exact cause, whether the solution failed or there are other undesirable numerical phenomena such as 'large step' divergence or 'curvature reversals' [13].

The convex optimization based trajectory generation has been accomplished using local optimization for finding an instantaneous optimal solution. These methods have the advantage of being versatile (they can, for instance, trade off different objectives such as error convergence or actuator input) and can rely on existing efficient convex optimization packages. A comprehensive analysis of trajectory generation for robot manipulator using closed loop inverse kinematics algorithm in a convex optimization framework is presented in [14]. It is centered on the Jacobian pseudoinverse and null space. The solution is achieved using gradient projection based methods. Trajectory generation is formulated using nonlinear optimization framework using Quadratic Programming in [15], [16]. The neural network based optimization framework for real-time trajectory generation is presented in [17]–[21].

The optimization techniques based on optimal control theory mainly aims to solve the trajectory generation problem based on Dynamic Programming. The theory of dynamic programming gives rise to HJB equation which is a necessary and sufficient condition for optimality of a control

Author * is with Cognitive Robotics, TU Delft while the work was carried out at IIT Kanpur, India. Author † is with IIT Jodhpur, India. Author ‡ is with IIT Kanpur, India.
e-mail: {r.prakash-1@tudelft.nl, jayant@iitj.ac.in, lbehera@iitk.ac.in}

solution with respect to a loss function. However, solving the HJB equation still remains a major hurdle. Therefore, several approximate solutions to the HJB equation have been proposed [22] and referred to as adaptive or approximate dynamic programming (ADP). The trajectory generation for a robot manipulator using single network adaptive critic under ADP framework is accomplished in [23], [24]. In these schemes, the theoretical convergence to the optimal cost with guarenteed stability proof is not shown. However, The near optimal convergence is shown using repeated training and extensive simulations. There exists very few works in the literature that has shown the convergence to the optimal cost in an analytical manner along with the proof of stability. Recently in [25], a Lyapunov stability based optimal ADP control framework for trajectory generation is proposed. The online neural network approximator learns the infinite-horizon cost function related to error dynamics in continuous time and calculates the corresponding optimal joint angle update law to minimize the cost function forward in time. To obtain a real-time closed loop solution has been addressed very recently in [26]. However, the solution for the online optimal trajectory generation scheme is proposed in free space without any obstacle. The solution uses SVD scheme which is unduly expensive computationally and not amenable to real-time extension to obstacle ridden environment. In this paper a comprehensive solution to the problem of optimal trajectory generation in obstacle ridden environments using simple matrix computation is proposed. The extensive comparison with the state-of-the-art optimization based schemes for same task in obstacle ridden environment is carried to demonstrate the significance of the proposed method. The closed form analytical HJB solution is shown to be more significant than the approximate optimal solutions for a complex and non-linear cost.

In the next section, the problem formulation for optimal control is presented. In section III and IV detailed mathematical derivation for the control scheme along with stability proof is presented. In section V, experimental results of the proposed control scheme for a 6-DOF robotic manipulator are evaluated with comparison with the state-of-the-art kinematic control solutions. This paper is finally concluded with prospects of future works in Section VI.

II. Background and Problem Formulation

A. Robot Manipulator Kinematics

Let $\mathbf{f}(\boldsymbol{\theta}) : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be the robot manipulator forward kinematics $(n \geq m)$ where $\boldsymbol{\theta}$ is the $\mathbb{R}^n$ joint position vector of the robot manipulator. The pose of the end-effector is given by

$$\mathbf{X}(\boldsymbol{\theta}) = \mathbf{f}(\boldsymbol{\theta}). \tag{1}$$

The representation of orientation in end-effector pose of the robot manipulator is non-trivial which significantly affects the calculation of pose error. For example, the popular "roll-pitch-yaw" representation are generally problematic and geometrically meaningless while calculating pose error. Therefore, In this paper, the end-effector pose and pose error $\in SE(3)$ are represented by a task space vector data structure to handle storage and operations on spatial frames. The task space vector is a data structure that keeps track of $SO(3)$ subgroups within the stored vector. The operations on this vector then implement the Lie group algebra to correctly compute addition or substraction in the task space. The embedding of the task space vector is done by combining three position coordinates $x, y, z \in \mathbb{R}$ and a sub vector containing a $SO(3)$ rotation represented as a unit quaternion. The substraction calculation of two task space vectors $\mathbf{X_1}$ and $\mathbf{X_2}$ first converts the quaternion into rotation matrices $\mathbf{R_1}$ and $\mathbf{R_2}$ and performs the roatation operation $\mathbf{R_2}^{-1}\mathbf{R_1}$. The result is then converted into $\omega_x, \omega_y, \omega_z$ and packed into output vector $\Delta\mathbf{X} = \mathbf{X_1} - \mathbf{X_2}$. Notice that the dimensionality of $\Delta\mathbf{X} \in \mathbb{R}^6$ and $\mathbf{X_1}, \mathbf{X_2} \in \mathbb{R}^7$ are different.

The time derivative of the kinematic model (1) yields the forward kinematic mapping from joint space to the operational space at the velocity level as $\dot{\mathbf{X}}(\boldsymbol{\theta}) = \frac{\partial \mathbf{f}}{\partial \boldsymbol{\theta}}\dot{\boldsymbol{\theta}}(t) = \mathbf{J}(\boldsymbol{\theta})\dot{\mathbf{q}}(t)$ where $\mathbf{J}(\boldsymbol{\theta})$ is the geometric Jacobian matrix [27]. In general, this matrix loses rank at singular configurations. Since the focus of this paper is mainly on the optimal control problem, $\mathbf{J}(\boldsymbol{\theta})$ is assumed to be of full row rank $(\text{rank} = m)$.

In this paper, we aim to design the joint velocities control law for trajectory generation of a robot manipulator. Then the robot manipulator kinematics can be rewritten by replacing $\dot{\boldsymbol{\theta}}(t)$ by $\mathbf{u}(t)$:

$$\dot{\mathbf{X}}(\boldsymbol{\theta}) = \mathbf{J}(\boldsymbol{\theta})\mathbf{u}(t), \text{ where } \mathbf{u}(t) = \dot{\boldsymbol{\theta}}(t). \tag{2}$$

B. Collision Cost Metric

Let there be O obstacles in the $m-$dimensional operational space of the robot manipulator and a virtual collision point $\mathbf{p}_j (j \in O)$ attached to each of them. Let us define collision error between the robot end effector and the j^{th} collision obstacle as $\mathbf{e}_{\mathbf{p}_j}(\boldsymbol{\theta}) = \mathbf{X}(\boldsymbol{\theta}) - \mathbf{p}_j$. Then the collision cost function $C_o(\boldsymbol{\theta})$ is given by $C_o(\boldsymbol{\theta}) = \sum_j^O we^{-k\left\|\mathbf{e}_{\mathbf{p}_j}(\boldsymbol{\theta})\right\|}$ where, the precision parameter k can be used to adjust the fall-off of the cost function, e.g. a precision factor of 50 will result in negligible cost when the obstacles are further than 10 cm apart from the end effector and the weighting factor w determines the maximum weightage of the collision cost when the distance of the end effector from the obstacle is zero.

III. Case I: Fixed End Pose

In this section, we shall consider the optimal regulation of $\mathbf{X}(\boldsymbol{\theta})$ to a constant desired value $\mathbf{X_r}$ in the m-dimensional operational space for the control of a robot manipulator in the presence of obstacles.

The derivative of the reference trajectory is given by $\dot{\mathbf{X}}_\mathbf{r} = \mathbf{0}$ Let the regulation error $\mathbf{e}(\boldsymbol{\theta}) = \mathbf{X}(\boldsymbol{\theta}) - \mathbf{X_r}$. Then its derivative w.r.t. time t is

$$\dot{\mathbf{e}}(\boldsymbol{\theta}) = \dot{\mathbf{X}}(\boldsymbol{\theta}) - \dot{\mathbf{X}}_\mathbf{r} = \mathbf{J}(\boldsymbol{\theta})\mathbf{u}_\mathbf{e}(t), \ \mathbf{u}_\mathbf{e}(t) = \mathbf{u}(t) \tag{3}$$

Considering the above system error dynamics (3) with $\boldsymbol{\theta}(t)$ as the system states, the optimal control law is derived next.

The infinite horizon HJB cost function for (3) is defined as

$$V(\boldsymbol{\theta}(t)) = \int_t^{\infty} L(\boldsymbol{\theta}(\tau), \mathbf{u_e}(\tau) d\tau \tag{4}$$

$$\text{where, } L(\boldsymbol{\theta}(t), \mathbf{u_e}(t)) = \frac{1}{2}(\mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e}(\boldsymbol{\theta}) + \mathbf{u_e}(t)^\intercal \mathbf{R} \mathbf{u_e}(t) + C_o(\boldsymbol{\theta})) \tag{5}$$

with $\mathbf{Q}$ as a $m \times m$ positive definite diagonal matrix to design the rate of convergence of the error and $\mathbf{R}$ as a $n \times n$ positive definite diagonal matrix which penalizes the inputs to optimize the cost function. Our goal is to find an optimal control law $\mathbf{u_e}(t)$ which minimizes the aforementioned cost function.

Then the hamiltonian for the cost function (4) with an associated admissible control input $\mathbf{u_e}(t)$ is written as

$$H(\boldsymbol{\theta}(t), \mathbf{u_e}(t)) = L(\boldsymbol{\theta}(t), \mathbf{u_e}(t)) + \frac{dV(\boldsymbol{\theta}(t))}{d\boldsymbol{\theta}} \dot{\boldsymbol{\theta}}(t) \tag{6}$$

where $\frac{d(V(\boldsymbol{\theta}(t)))}{d\boldsymbol{\theta}}$ is $1 \times$ n vector.

The optimal control input that minimizes the cost function (4) also minimizes the hamiltonian (6); therefore optimal control is found by solving the stationarity condition $\frac{\partial H(\boldsymbol{\theta}(t), \mathbf{u_e}(t))}{\partial \mathbf{u_e}(t)} = 0$ *i.e.* $\min_{\mathbf{u_e}} \left(L(\boldsymbol{\theta}(t), \mathbf{u_e}(t)) + \frac{dV^*}{d\boldsymbol{\theta}} \dot{\boldsymbol{\theta}}(t) \right) = 0$. Putting the expressions for $L(\mathbf{e}(t), \mathbf{u_e}(t))$ and $\dot{\boldsymbol{\theta}}(t)$ from eqs. (5) and (2), respectively,

$$\min_{\mathbf{u_e}} \left(\frac{1}{2} \big(\mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e}(\boldsymbol{\theta}) + \mathbf{u_e}(t)^\intercal \mathbf{R} \mathbf{u_e}(t) + C_o(\boldsymbol{\theta}) \big) + \frac{dV^*}{d\boldsymbol{\theta}} \mathbf{u_e}(t) \right) = 0 \tag{7}$$

Differentiating (7) with respect to $\mathbf{u_e}$, we get

$$\mathbf{u_e}^*(t) = -\mathbf{R}^{-1} \left(\frac{dV^*}{d\boldsymbol{\theta}} \right)^\intercal \tag{8}$$

In order to find the expression for $\left(\frac{dV^*}{d\boldsymbol{\theta}} \right)$, we put the optimal $\mathbf{u_e}$ from (8) in (7),

$$\min_{\mathbf{u_e}} \left(\frac{1}{2} \mathbf{e}(\boldsymbol{\theta})(t)^\intercal \mathbf{Q} \mathbf{e}(\boldsymbol{\theta})(t) + \frac{1}{2} \left(\frac{dV^*}{d\boldsymbol{\theta}} \mathbf{R}^{-1} \frac{dV^{*\intercal}}{d\boldsymbol{\theta}} \right) + \frac{1}{2} C_o(\boldsymbol{\theta}) + \frac{dV^*}{d\boldsymbol{\theta}} (-\mathbf{R}^{-1} \frac{dV^{*\intercal}}{d\boldsymbol{\theta}}) \right) = 0 \tag{9}$$

$$\left(\frac{dV^*}{d\boldsymbol{\theta}} \right) \mathbf{R}^{-1} \left(\frac{dV^*}{d\boldsymbol{\theta}} \right)^\intercal = \mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e}(\boldsymbol{\theta}) + C_o(\boldsymbol{\theta}) \tag{10}$$

This is a quadratic equation of the form $\mathbf{Y}^\intercal \mathbf{Y} = x$, where $\mathbf{Y} \in \mathbb{R}^m$ and $x \in \mathbb{R}$. It has a solution of the form $\mathbf{Y} = \sqrt{x} \boldsymbol{\Phi}$, where $\boldsymbol{\Phi} \in \mathbb{R}^m$ is a vector with $\boldsymbol{\Phi}^\intercal \boldsymbol{\Phi} = 1$. Thus, we get

$$\mathbf{R}^{-\frac{1}{2}} \left(\frac{dV^*}{d\boldsymbol{\theta}} \right)^\intercal = \sqrt{\mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e}(\boldsymbol{\theta}) + C_o(\boldsymbol{\theta})} \boldsymbol{\Phi} \tag{11}$$

Substituting (11) in (8) gives

$$\mathbf{u_e}^*(t) = -\mathbf{R}^{-\frac{1}{2}} \boldsymbol{\Phi} \sqrt{\mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e}(\boldsymbol{\theta}) + C_o(\boldsymbol{\theta})} \tag{12}$$

. The expression for $\boldsymbol{\Phi}$ is determined using Lyapunov Stability criteria presented next.

A. Stability Proof

Proof. The stability of the system can be analyzed with the help of a Lyapunov function defined as $\mathcal{V}(\mathbf{e}(\boldsymbol{\theta})) = \frac{1}{2} \mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{e}(\boldsymbol{\theta})$. Then the time derivative of the Lyapunov function is $\dot{\mathcal{V}}(\mathbf{e}(\boldsymbol{\theta})) = \mathbf{e}(\boldsymbol{\theta})^\intercal \dot{\mathbf{e}}(\boldsymbol{\theta}) = \mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{J}(\boldsymbol{\theta}) \mathbf{u_e}^*(t)$

$$= -\mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{J}(\boldsymbol{\theta}) \mathbf{R}^{-\frac{1}{2}} \boldsymbol{\Phi} \sqrt{\mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e}(\boldsymbol{\theta}) + C_o(\boldsymbol{\theta})} \tag{13}$$

Now to ensure guarenteed system convergence $\dot{\mathcal{V}}(\mathbf{e}(\boldsymbol{\theta}))$ needs to be negative definite. Therefore, if $\boldsymbol{\Phi}$ is chosen as

$$\boldsymbol{\Phi} = \frac{\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e}(\boldsymbol{\theta})}{||(\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e}(\boldsymbol{\theta}))||} \tag{14}$$

satisfying the condition $\boldsymbol{\Phi}^\intercal \boldsymbol{\Phi} = 1$, then

$$\dot{\mathcal{V}}(\mathbf{e}(\boldsymbol{\theta})) = -\mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{J}(\boldsymbol{\theta}) \mathbf{R}^{-\frac{1}{2}} \frac{\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e}(\boldsymbol{\theta})}{||(\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e}(\boldsymbol{\theta}))||} \sqrt{\mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e}(\boldsymbol{\theta}) + C_o(\boldsymbol{\theta})} \tag{15}$$

It can be concluded that $\dot{\mathcal{V}}(\mathbf{e}(\mathbf{q}))$ becomes negative definite and regulation error converges to zero asymptotically implying optimality and guarenteed convergence concurrently. □

B. Stabilizing Control Law

Finally we get the stabilizing optimal control by combining (12) and (14).

$$\mathbf{u_e}^*(t) = -\mathbf{R}^{-\frac{1}{2}} \frac{\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e}(\boldsymbol{\theta})}{||(\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e}(\boldsymbol{\theta}))||} \sqrt{\mathbf{e}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e}(\boldsymbol{\theta}) + C_o(\boldsymbol{\theta})} \tag{16}$$

IV. Case II : Time Varying End Pose Trajectory

In this section, we shall consider the tracking of $\mathbf{X}(\boldsymbol{\theta})$ to a time varying end pose reference trajectory $\mathbf{X_r}(t)$ in the m-dimensional operational space a robot manipulator. The optimal tracker is considered as an extension to regulation problem [22]. The objective for the infinite time optimal tracking control problem is to design the optimal control law $\mathbf{u}^*(t)$ to ensure that the nonlinear system (1) tracks a time varying desired trajectory $\mathbf{X_r}(t)$ with known time derivative, $\dot{\mathbf{X}}_\mathbf{r}(t)$, in an optimal manner. It is observed that the optimal control input for the case of dynamic tracking consists of an optimal feedback term $\mathbf{u_{te}}$ and a predetermined feedforward term used to compensate to satisfy the steady state velocity requirement [28]. To ensure that the desired velocity is reached during steady state, a feedforward control input, $\mathbf{u_{ss}}(t)$ is found using $\dot{\mathbf{X}}_\mathbf{r}(t) = \mathbf{J}(\boldsymbol{\theta}) \mathbf{u_{ss}}(t)$.

Now, because of the redundancy of the robot manipulator, unique solution for such a $\mathbf{u_{ss}}(t)$ does not exist. Hence, the solution which minimizes the kinetic energy is chosen i.e., $\mathbf{u_{ss}}(t)$ that minimizes $\mathbf{u_{ss}}(t)^\intercal \mathbf{u_{ss}}(t)$. The solution for such a $\mathbf{u_{ss}}(t)$ can be found as $\mathbf{u_{ss}}(t) = \mathbf{J}(\boldsymbol{\theta})^\dagger \dot{\mathbf{X}}_\mathbf{r}(t)$, where $\mathbf{J}(\boldsymbol{\theta})^\dagger = \mathbf{J}(\boldsymbol{\theta})^\intercal (\mathbf{J}(\boldsymbol{\theta}) \mathbf{J}(\boldsymbol{\theta})^\intercal)^{-1}$ is the pseudo-inverse of $\mathbf{J}(\boldsymbol{\theta})$.

A. Tracking Control law

Let the tracking error $\mathbf{e_t}(\boldsymbol{\theta}) = \mathbf{X}(\boldsymbol{\theta}) - \mathbf{X_r}(t)$.

The optimal tracking control law can then be obtained by $\mathbf{u}^*(t) = \mathbf{u_{te}}^*(t) + \mathbf{u_{ss}}(t)$

$$= -\mathbf{R}^{-\frac{1}{2}} \frac{\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e_t}(\boldsymbol{\theta})}{||(\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e_t}(\boldsymbol{\theta}))||} \sqrt{\mathbf{e_t}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e_t}(\boldsymbol{\theta})} + \mathbf{J}(\boldsymbol{\theta})^\dagger \dot{\mathbf{X}}_\mathbf{r}(t) \tag{17}$$

B. Stability Proof

Proof. The time derivative of the tracking error is given by

$$\begin{aligned} &\dot{\mathbf{e_t}}(\boldsymbol{\theta}) = \dot{\mathbf{X}}(\boldsymbol{\theta}) - \dot{\mathbf{X}}_\mathbf{r}(t) = \mathbf{J}(\boldsymbol{\theta})\mathbf{u}^*(t) - \dot{\mathbf{X}}_\mathbf{r}(t) \\ &= \mathbf{J}(\boldsymbol{\theta})(-\mathbf{R}^{-\frac{1}{2}} \frac{\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e_t}(\boldsymbol{\theta})}{||(\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e_t}(\boldsymbol{\theta}))||} \sqrt{\mathbf{e_t}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e_t}(\boldsymbol{\theta})} \\ &+ \mathbf{J}(\mathbf{q})^\dagger \dot{\mathbf{X}}_\mathbf{r}(t)) - \mathbf{J}(\boldsymbol{\theta})\mathbf{J}(\boldsymbol{\theta})^\intercal (\mathbf{J}(\boldsymbol{\theta})\mathbf{J}(\boldsymbol{\theta})^\intercal)^{-1} \dot{\mathbf{X}}_\mathbf{r}(t) \\ &= -\mathbf{J}(\boldsymbol{\theta})\mathbf{R}^{-\frac{1}{2}} \frac{\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e_t}(\boldsymbol{\theta})}{||(\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e_t}(\boldsymbol{\theta}))||} \sqrt{\mathbf{e_t}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e_t}(\boldsymbol{\theta})} \end{aligned} \tag{18}$$

Consider the following lyapunov function $\mathcal{V}_t(\mathbf{e_t}(\boldsymbol{\theta})) = \frac{1}{2}\mathbf{e_t}(\boldsymbol{\theta})^\intercal \mathbf{e_t}(\boldsymbol{\theta})$. Then the time derivative of $\mathcal{V}_t(\mathbf{e_t}(\boldsymbol{\theta}))$ can be written as $\dot{\mathcal{V}}_t(\mathbf{e_t}(\boldsymbol{\theta})) = \mathbf{e_t}(\boldsymbol{\theta})^\intercal \dot{\mathbf{e_t}}(t)$

$$= -\mathbf{e_t}(\boldsymbol{\theta})^\intercal \mathbf{J}(\boldsymbol{\theta})\mathbf{R}^{-\frac{1}{2}} \frac{\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e_t}(\boldsymbol{\theta})}{||(\mathbf{J}(\boldsymbol{\theta})^\intercal \mathbf{e_t}(\boldsymbol{\theta}))||} \sqrt{\mathbf{e_t}(\boldsymbol{\theta})^\intercal \mathbf{Q} \mathbf{e_t}(\boldsymbol{\theta})}$$

It can be concluded that $\dot{\mathcal{V}}(\mathbf{e_t}(\boldsymbol{\theta}))$ becomes negative definite and tracking error converges to zero asymptotically. □

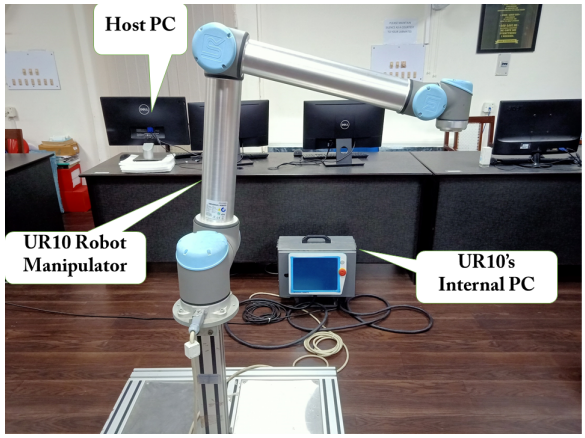


Fig. 1: Hardware Setup

V. Experimental Results

In this section, we present the details of realtime experimental validations of the proposed HJB control scheme along with similar approximate HJB based ADP control from the literature. To demonstrate the performance and efficacy of the schemes a 6-DOF UR 10 robot manipulator is employed as testbed.

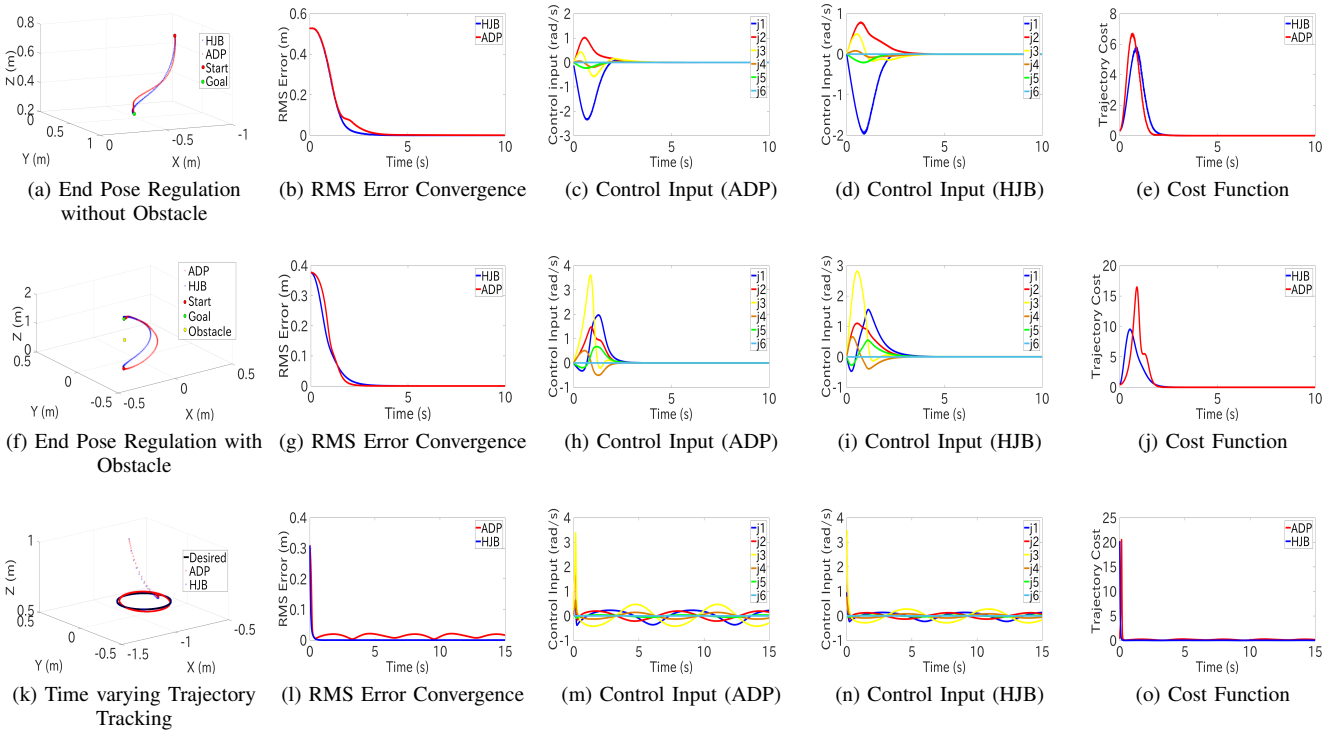
A. Experimental Setup

In order to evaluate the control performance of the proposed controller, a UR10 robotic manipulator with six joints is employed as a test bed. The structure of this robotic manipulator is shown in Figure (1). It consists of a UR10 robot manipulator with its internal PC and a host PC/external computer. It exposes its real-time interface for joint velocity control schemes using URScript commands. The realtime interface runs on the UR10's internal PC using UrDriver wrapper class integrated with ROS with inner control loop frequency operating at $125Hz$. The main controller runs on Host PC which publishes the generated control commands to the robot's real-time interface at 125 Hz over ethernet in a ROS framework.

B. Case I: Fixed End Pose

1) Without Obstacles: Experimental results are presented to demonstrate the performance of the proposed optimal control of a robot manipulator for the task of end pose regulation in case of obstacle free robot workspace. The objective is to actuate the robot manipulator using optimal joint velocities to the individual robot joints which is obtained by minimizing the cost function (4) with $C_o(\mathbf{e}(t)) = 0$, such that the robot end effector reaches a target pose ($\mathbf{X_r}$) in the cartesian space from any given start pose. The experimental results generated from a typical run using the proposed control with start pose (position : $[-0.7, 0.3, 0.65]m$, orientation :$[-0.500, 0.512, 0.500, 0.487]$) to a fixed target pose (position : $[-0.5, 0.5, 0.2]m$, orientation :$[-0.500, 0.512, 0.500, 0.487]$) in the cartesian space is shown in Figure (2(a,b,d,e)). The controller gain parameters during the run is diag($\mathbf{Q}$)=$[10^6; 10^6; 10^6; 10^5; 10^5; 10^5]$ and diag($\mathbf{R}$)=[0.5 ;0.5 ;0.5 ;1 ;1 ;1]. For a qualitative comparison purpose, a ADP control approach [25] was utilized to solve the end pose regulation control problem for the same start and target pose using the same cost function (4) with $C_o(\mathbf{e}(t)) = 0$. This controller also generates velocity control inputs to actuate individual robot joints such that the regulation control error converges to zero in a short transient time. The results are shown in Figure (2(a,b,c,e)). The controller gains in [25] were tuned to obtain similar time domain performance as that in the previous case.

2) With Obstacles: Experimental results are presented to demonstrate the performance of the proposed optimal control of a robot manipulator for the task of end pose regulation in presence of obstacle in the robot workspace. For this, the robot manipulator is actuated using optimal joint velocities to the individual robot joints which is obtained by minimizing the cost function (4), such that the robot end effector reaches a target pose ($\mathbf{X_r}$) in the cartesian space from any given start pose while avoiding obstacles in the workspace. The collision weighting factor w and the precision parameter k is selected as 5 and 50 respectively such that the collision cost is maximum at 5 when the end effector is very near to the obstacle and and it falls off exponentially to 0 when the end effector is farther than a radius of 10 cm from the obstacle. Also the obstacles are always farther than a minimum of 10 cm from the goal point. The experimental results generated from a typical run with start pose (position : [-0.25,-0.25,0.1]m, orientation :[-0.053, 0.034, 0.704, 0.707]) to a fixed target pose (position : [0,0.1,0.4]m, orientation :$[-0.053, 0.034, 0.704, 0.707]$) in the cartesian space



(a) End Pose Regulation without Obstacle (b) RMS Error Convergence (c) Control Input (ADP) (d) Control Input (HJB) (e) Cost Function

(f) End Pose Regulation with Obstacle (g) RMS Error Convergence (h) Control Input (ADP) (i) Control Input (HJB) (j) Cost Function

(k) Time varying Trajectory Tracking (l) RMS Error Convergence (m) Control Input (ADP) (n) Control Input (HJB) (o) Cost Function

Fig. 2: Experimental results for end pose regulation control without obstacles (a-e), with obstacles (f-j) and time varying trajectory tracking for a time varying elliptical trajectory (k-o) using proposed control scheme and ADP control [25].

TABLE I: Quantitative test comparison of the proposed control with the state-of-the-art optimal control algorithms for real-time trajectory generation.

Scheme	Class	Proof	Case I without obstacles	Case I with obstacles	Case II ($\omega < 1rad/s$)	Case II ($\omega > 1rad/s$)
			ANTC1	ANTC1	ANTC1	ANTC1
RRT+ KDL [7]	Numerical, Local, Newton method	No	880 ±20.4	3649±85.27	93±4.85	116±7.54
SVF CLIK [14]	Instantaneous, Local, Gradient Projection	No	675±17.2	2816±55.75	82±2.4	98±3.58
STOMP [19]	Probabilistic, Local, Gradient Descent	No	572±14.3	2337±40.97	70±1.9751	87±2.17
RNN [20]	Convex, Local, KKT conditions	Yes	525±6.2	1669±35.5	61±1.5	71±1.64
ADP [25]	Sub-optimal, Global, Approximate HJB	Yes	250±2	780±15.25	47±1.05	55±1.27
Proposed HJB	Optimal, Global, Exact HJB	Yes	225±1.5	450±9.77	35±0.5	42±0.59

with an obstacle at (position : $[-0.125, -0.175, 0.25]m$) is shown in Figure (2(f,g,i,j)). The controller gain parameters during the run is diag(Q)=$[10^6; 10^6; 10^6; 10^5; 10^5; 10^5]$ and diag(R)=$[0.25; 0.25; 0.25; 0.5; 0.5; 0.5]$. The ADP control approach [25] was utilized to solve the end pose regulation control problem for the same start and target pose with the obstacle using the same cost function (4). The results are shown in Figure (2(f,g,h,j)). The controller gains in [25] were tuned to obtain similar time domain performance as that in the previous case.

C. Case II: Time Varying End Pose Trajectory

In this section, the tracking control of a time varying reference trajectory given by the control law (17) is validated through experiments. Many circular time varying reference trajectory was generated using sampled parameters like center of the circle, radius of the circle and speed of the circle. A typical experimental run generated with a sampled start pose and tracking a time varying reference trajectory moving at an angular speed of 1.5 rad/s along a circle centered at $[-0.8, 0, 0.5]m$ with a radius of $0.2m$ is shown in Figure (2k). The controller gain parameters during the run is diag($\mathbf{Q}$)=$[10^6\ 10^6\ 10^6\ 10^5\ 10^5\ 10^5]$ and diag($\mathbf{R}$)=$[0.25\ 0.25\ 0.25\ 0.5\ 0.5\ 0.5]$. The results of the trajectory tracking using ADP control for the same problem setup above is shown in Figure(2(k,l,m,o)).

D. Quantitative Comparison Test Methodology

In order to quantify the performance of the proposed optimal control for real-time trajectory generation, an automated test process was used for comparing various class of optimization based solution from the literature.

The quantitative test methodology for comparing the proposed method for optimal control against the various solutions mentioned in Table I is entailed next.

First the test samples are generated over robot's task workspace in case of both regulation and tracking problem. The sample space consists of feasible robot end poses within robot's reach. A total of 25 target poses were generated in case of end pose regulation with 5 variable start poses. In case of end pose regulation control with obstacles, an obstacle is placed in midway of the line joining the start and goal poses. For the case of time varying trajectory tracking performance in the robot's task space, 25 different feasible time varying circular reference trajectories with varying parameters (radius of the circle, center of the circle and speed of the circle) were generated in the workspace of the robot.

Then the above mentioned control schemes for real-time trajectory generation are tested on those test points over all workspace and compared on a metric defined as Average Normalised Trajectory Cost (ANTC). The ANTC is defined as average normalised trajectory cost over all trajectories. The normalized cost for i^{th} trajectory is given by $C_i = C/N$, where N is the number of total time instances until the error converges to a threshold in case of end pose regulation and total time of execution in case of trajectory tracking. Hence the average normalized trajectory cost over all trajectory is given by $\frac{\sum_i^{N_t} V_i}{N_t}$, where N_t is total number of trajectories. Now we define the ANTC metric corresponding to trajectory cost $C = \mathbf{e}(t)^\intercal \mathbf{e}(t) + \mathbf{u}(t)^\intercal \mathbf{u}(t) + \dot{\mathbf{u}}(t)^\intercal \dot{\mathbf{u}}(t)$. This metric is a measure of accuracy and optimal effort with smooth trajectories. To ensure fairness in comparison the following methodology was adopted. First all the algorithms are run for the same problem setup using best selection of controller gain parameters until satisfactory task error performance is achieved (e.g. the RMS regulation error or tracking error converges below 10^{-3}). Then the controller gain parameter of each of the controller is tweaked to match the same time domain convergence while ensuring that the maximum control limits of the individual joints is not violated in all the cases. So, now for the same time domain task performance, both the control methods are compared on a same metric whose physical significance is that how much control effort is invested for the same accuracy of the task and also how smooth are the trajectory profiles. The smaller value of these metrics would imply smooth and optimal control for the same robot motion tasks. It would also ensure optimal actuator torque input in the secondary/low level servo control loop for the same task performance in the real world applications.

The above experiment was done for 10 trials. The Table I summarizes the comparison drawn on the basis of metric ANTC with their mean and variance. The comparision has been arranged as per their class of optimization and methodology of solution. In essence the performance of schemes based on optimal control theory involving solution of HJB equation performs better than other classes of optimization algorithms. The study reflects that among the convex optimization formulations of the same problem, the RNN based optimization using KKT optimization criteria performs slightly better than its probabilistic counterpart STOMP along with the analytic proof of stability. Among the global optimization algorithms involving solution of HJB equation, the important observation from the comparison test is that the proposed controller performs significantly better than the ADP control in terms of ANTC for regulation control with obstacles and trajectory tracking accuracy for reference trajectories with high speed. This implies accurate, optimal and smoother trajectory profiles using the proposed HJB based solution. Also the tracking accuracy of the ADP controller shows oscillatory behaviour in realtime experiments with reference trajectories having higher speed ($\omega \geq 1rad/s$), see Figure (2l) for $\omega = 1.5rad/s$. The observed bahaviour may be accounted due to the iterative update of neural network weights towards optimality which is not fast enough in case of high speed tracking. Therefore, the conclusion of the analysis is that the analytic closed form HJB solution shall significantly perform better than the approximate NN solution when the cost function becomes complex and more non-linear.

VI. Warehousing Applications

To demonstrate the significance of proposed real-time trajectory generation scheme for real-world robotics application, a demo for warehouse automation case is included in this paper. In case I, the robot manipulator pick desired items from a table in the presence of obstacles and place it in a storage box and in case II, the robot manipulator has to pick the items from a moving conveyor belt. The task space target locations of items is integrated with the proposed control scheme through real-time visual feedback. The real-time experimental demo of warehousing application has been included as supplementary video submission and available online at [29].

VII. Conclusion and Future Works

In this work, we have designed an optimal controller for real-time trajectory generation of a robot manipulator using HJB framework. This is a necessary and sufficient condition for optimality of a control solution with respect to a cost function. The control input update law is derived using a closed form solution of HJB equation in contrast to ADP based approximate HJB solution in the state-of-the-art. The stability analysis of the proposed control scheme is guarenteed using Lyapunov stability. Both the problem of end pose regulation and time varying trajectory tracking in the task space is solved under the framework of optimal control. Using the proposed optimal end pose regulation and time varying trajectory tracking control law (16), (17), it has been experimentally demonstrated that robot performs desired tasks with optimal cost as evident from Table 1. The proposed controller has the advantage of better cost with steady state trajectory tracking accuracy for reference trajectories with high speed than ADP based controller [25] when the cost function becomes more complex and non-linear. The collison cost metric defined in this paper considers a

spherical static obstacles in the robot task space. This is one of the limitations of this work and a possible opportunity to extend it for different geometry/shape/size of the obstacles and also time varying obstacles. The collision cost needs to be also defined in robot configuration space instead of only end-effector. Instead of eye-to-hand visual feedback, the proposed controller can be integrated with visual feedback in eye-on-hand to develop optimal visual servoing algorithms where the robot jacobian would be extended with the image jacobian.

REFERENCES

[1] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[2] S. Morgan and M. S. Branicky, "Sampling-based planning for discrete spaces," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 2. IEEE, 2004, pp. 1938–1945.

[3] J. Lin, N. Somani, B. Hu, M. Rickert, and A. Knoll, "An efficient and time-optimal trajectory generation approach for waypoints under kinematic constraints and error bounds," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 5869–5876.

[4] A. Goldenberg, B. Benhabib, and R. Fenton, "A complete generalized solution to the inverse kinematics of robots," *IEEE Journal on Robotics and Automation*, vol. 1, no. 1, pp. 14–20, 1985.

[5] J. Angeles, "On the numerical solution of the inverse kinematic problem," *The International Journal of Robotics Research*, vol. 4, no. 2, pp. 21–37, 1985.

[6] K. K. Hauser, "Fast interpolation and time-optimization on implicit contact submanifolds." in *Robotics: Science and systems*, vol. 145. Citeseer, 2013.

[7] R. Smits, "KDL: Kinematics and Dynamics Library," http://www.orocos.org/kdl.

[8] A. Goldenberg, J. Apkarian, and H. Smith, "A new approach to kinematic control of robot manipulators," *Journal of dynamic systems, measurement, and control*, vol. 109, no. 2, pp. 97–103, 1987.

[9] Y. T. Tsai and D. E. Orin, "A strictly convergent real-time solution for inverse kinematics of robot manipulators," *Journal of Robotic Systems*, vol. 4, no. 4, pp. 477–501, 1987.

[10] L.-C. Wang and C.-C. Chen, "A combined optimization method for solving the inverse kinematics problems of mechanical manipulators," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 4, pp. 489–499, 1991.

[11] E. Sariyildiz and H. Temeltas, "Performance analysis of numerical integration methods in the trajectory tracking application of redundant robot manipulators," *International Journal of Advanced Robotic Systems*, vol. 8, no. 5, p. 63, 2011.

[12] D. A. Drexler and L. Kovács, "Second-order and implicit methods in numerical integration improve tracking performance of the closed-loop inverse kinematics algorithm," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2016, pp. 003 362–003 367.

[13] D. Whitney, "Optimum step size control for newton-raphson solution of nonlinear vector equations," *IEEE Transactions on Automatic Control*, vol. 14, no. 5, pp. 572–574, 1969.

[14] A. Colomé and C. Torras, "Closed-loop inverse kinematics for redundant robots: Comparative assessment and two enhancements," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 2, pp. 944–955, April 2015.

[15] O. Kanoun, F. Lamiraux, and P. Wieber, "Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 785–792, Aug 2011.

[16] P. Beeson and B. Ames, "Trac-ik: An open-source library for improved solving of generic inverse kinematics," in *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*. IEEE, 2015, pp. 928–935.

[17] S. Li, Y. Zhang, and L. Jin, "Kinematic control of redundant manipulators using neural networks," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2243–2254, 2017.

[18] S. Li, Z. Shao, and Y. Guan, "A dynamic neural network approach for efficient control of manipulators," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 5, pp. 932–941, May 2019.

[19] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 4569–4574.

[20] J. Liang, Z. Xu, X. Zhou, S. Li, and G. Ye, "Recurrent neural networks-based collision-free motion planning for dual manipulators under multiple constraints," *IEEE Access*, vol. 8, pp. 54 225–54 236, 2020.

[21] R. Prakash, L. Behera, S. Mohan, and S. Jagannathan, "Dynamic trajectory generation and a robust controller to intercept a moving ball in a game setting," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 4, pp. 1418–1432, 2020.

[22] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal control*. John Wiley & Sons, 2012.

[23] R. Padhi, N. Unnikrishnan, X. Wang, and S. Balakrishnan, "A single network adaptive critic (snac) architecture for optimal control synthesis for a class of nonlinear systems," *Neural Networks*, vol. 19, no. 10, pp. 1648–1660, 2006.

[24] P. K. Patchaikani, L. Behera, and G. Prasad, "A single network adaptive critic-based redundancy resolution scheme for robot manipulators," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 8, pp. 3241–3253, 2012.

[25] A. Menon, R. Prakash, and L. Behera, "Adaptive critic based optimal kinematic control for a robot manipulator," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 1316–1322.

[26] R. Prakash, L. Behera, S. Mohan, and S. Jagannathan, "Dual-loop optimal control of a robot manipulator and its application in warehouse automation," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 1, pp. 262–279, 2022.

[27] M. W. Spong and M. Vidyasagar, *Robot dynamics and control*. John Wiley & Sons, 2008.

[28] T. Dierks and S. Jagannathan, "Optimal control of affine nonlinear continuous-time systems," in *American Control Conference (ACC), 2010*. IEEE, 2010, pp. 1568–1573.

[29] R. Prakash, J. K. Mohanta, and L. Behera. Closed form hjb solution for path planning of a robot manipulator with warehousing applications. Youtube. [Online]. Available: https://youtu.be/CyGrPNL4tCI