2015 IEEE/SICE International Symposium on System Integration (SII)
December 11-13, 2015. Meijo University, Nagoya, Japan

Infeasibility-free Inverse Kinematics Method

Wael Suleiman[1], Fumio Kanehiro[2] and Eiichi Yoshida[3]

Abstract— **The problem of inverse kinematics is revisited in the present paper. The paper is focusing on the problem of solving the inverse kinematics problem while respecting velocity limits on both the robot's joints and the end-effector. Even-though the conventional inverse kinematics algorithms have been proven to be efficient in many applications, defining an admissible trajectory for the end-effector is still a burdensome task for the user, and the problem can easily become unsolvable. The main idea behind the proposed algorithms is to consider the sampling time as a free variable, hence adding more flexibility to the optimization problem associated with the inverse kinematics. We prove that the reformulated problem has always a solution if the end-effector path is in the reachable space of the robot, thus solving the problem of infeasibility of conventional inverse kinematics methods.**

To validate the proposed approach, we have conducted three simulations scenarios. The simulation results point that while the conventional inverse kinematics methods fail to track precisely a desired end-effector trajectory, the proposed algorithms always succeed.

I. Introduction

The inverse kinematics problem is one of the most studied problems in robotics, and many methods to solve it have been proposed in the literature. Those methods are extensively used in robotics, they can be also combined with other techniques, e.g. force control, to cope with the imperfect modelling of the robot and its dynamic parameters.

Generally speaking, the objective is to find a vector in the configuration space that satisfies constraints in the operational space, in other words the values of the robot's joints to make the end-effector reaches a desired goal (position and orientation) in the Cartesian space. Although in some special cases the problem can be solved analytically [1], [2], it is generally solved numerically [3], [4]. Many efficient and robust numerical algorithms have been proposed for solving the inverse kinematics problem [5], [6], [7], [3], [4], only to cite few.

The inverse kinematics problem was originally formulated as follows:

$$\min_{\dot{\boldsymbol{q}}_t} {\dot{\boldsymbol{q}}_t}^T Q \dot{\boldsymbol{q}}_t$$

subject to

$$J\dot{\boldsymbol{q}}_t = \dot{r}_t \tag{1}$$

where Q is a diagonal and positive semi-definite matrix, J is the Jacobian matrix, $\dot{\boldsymbol{q}} \in \mathbb{R}^n$ is the joint velocity and $\dot{r}_t$ is the linear and angular velocity of the end-effector.

The optimization problem (1) can be efficiently solved using the pseudo-inverse technique [3], [4] as follows:

$$\dot{\boldsymbol{q}}_t = \hat{J}^\dagger \dot{r}_t + (I - \hat{J}^\dagger J) z \tag{2}$$

where $\hat{J}^\dagger = Q^{-1} J^T \left(J\, Q^{-1}\, J^T \right)^{-1}$ and z is an arbitrary vector. If Q is the identity matrix, $\hat{J}^\dagger$ becomes the well-known Moore-Penrose pseudo-inverse. It is worth mentioning that the optimization problem (1) is time-invariant as the time derivative can be simply replaced by the difference. Thus, we obtain the following equivalent optimization problem:

$$\min_{\Delta \boldsymbol{q}} \Delta \boldsymbol{q}^T Q \Delta \boldsymbol{q}$$

subject to

$$J\Delta \boldsymbol{q} = \Delta r \tag{3}$$

The solution of the above problem is given by:

$$\Delta \boldsymbol{q}_t = \hat{J}^\dagger \Delta r_t + (I - \hat{J}^\dagger J)\hat{z} \tag{4}$$

However, in order to consider the velocity and joints limits as well as avoid obstacles, inequality constraints should be added. The general inverse kinematics problem can be therefore formulated as follows:

$$\min_{\dot{\boldsymbol{q}}_t} {\dot{\boldsymbol{q}}_t}^T Q \dot{\boldsymbol{q}}_t$$

subject to

$$\begin{aligned} J\dot{\boldsymbol{q}}_t &= \dot{r}_t \\ b^- \leq A\dot{\boldsymbol{q}}_t &\leq b^+ \\ \dot{\boldsymbol{q}}^- \leq \dot{\boldsymbol{q}}_t &\leq \dot{\boldsymbol{q}}^+ \end{aligned} \tag{5}$$

$\dot{\boldsymbol{q}}^-$ and $\dot{\boldsymbol{q}}^+$ are respectively the lower and upper limits of the velocity. A, b^- and b^+ represent the additional constraints that result from considering geometric constraints, such as collision avoidance, or physical constraints, such as joint limits, for more details the reader is referred to [8], [9], [10], [11], [12].

The main differences between (1) and (5) are:

- There is no closed-form expression for the solution of (5). However, it is a convex optimization problem if Q is a semi-definite positive matrix, and in this case, the problem can be efficiently solved by a Quadratic Programming (QP) solver [9], [10], [11], [12].

[1]Wael Suleiman is with Electrical and Computer Engineering Department, Faculty of Engineering, Université de Sherbrooke, 2500, boul. Universite, Sherbrooke, Quebec (Canada) J1K 2R1. Wael.Suleiman@USherbrooke.ca

[2]F. Kanehiro is with Intelligent Systems Research Institute, National Institute of Advanced Industrial Science and Technology (AIST), 1-1-1 Umezono, Tsukuba, Ibaraki 305-8568 Japan. f-kanehiro@aist.go.jp

[3]E. Yoshida is with CNRS-AIST, JRL (Joint Robotics Laboratory), UMI 3218/CRT, AIST, 1-1-1 Umezono Tsukuba Ibaraki, Japan. e.yoshida@aist.go.jp

- Problem (5) is time-variant.

A major difficulty that is faced, in practice, by the user is how to define the trajectory of end-effector (r_t). It is clear that if the end-effector moves fast, the joints velocities will be saturated and as a result the end-effector trajectory diverges from the desired one. A solution to the previous problem is to ensure that the end-effector velocity is slow enough to avoid the joints velocity saturation issue, however this solution is not the optimal one in practice, as it yields to a very slow motion and it does not take full advantage of the robot's physical limits.

A well known approach in the literature to deal simultaneously with the velocity limits of the joints and the end-effector is time parameterization [13], however this approach is mainly adapted for offline computation and it needs the whole path to be known in advance.

It is worth to point the difference between a path and a trajectory. A *path* denotes the locus of points in the joint space, or in the operational space, the robot has to follow in the execution of the desired motion, while a *trajectory* is a path on which a time law is specified [14].

In this paper, we are interested in a robust online implementation of inverse kinematics, which is infeasibility-free and can efficiently deal with the velocity limits of both the joints and the end-effector.

II. Problem Formulation

We consider two cases, the first one focuses on transforming the original inverse kinematics problem into an equivalent optimization problem which is always feasible, while the second focuses on incorporating the end-effector velocity limits into the inverse kinematics problem. The main idea is instead of supposing that the sampling time (T) fixed, making it a free variable, and solving the inverse kinematics problem as a function of $\Delta \boldsymbol{q}$ and T. Thus:

$$\dot{\boldsymbol{q}}_t \cong \frac{\Delta \boldsymbol{q}}{T} \tag{6}$$

The end-effector velocity vector is defined as follows:

$$\dot{r}_t = \begin{bmatrix} \boldsymbol{v}_t \\ \boldsymbol{\omega}_t \end{bmatrix} \tag{7}$$

where $\boldsymbol{v}_t$ is the linear velocity of the end-effector:

$$\boldsymbol{v}_t \cong \frac{\Delta X_e}{T} \tag{8}$$

$X_e \in \mathbb{R}^3$ is the Cartesian position of the end-effector.

$\boldsymbol{\omega}_t$ is the angular velocity of the end-effector and can be obtained using the following formula:

$$[\boldsymbol{\omega}_t]^\wedge \cong \frac{\Delta \boldsymbol{R}_e}{T} {\boldsymbol{R}_e}^T \tag{9}$$

where $\boldsymbol{R}_e$ is the rotation matrix of the end-effector frame, and $[.]^\wedge$ designs the skew operator defined as follows:

$$\begin{aligned} &[.]^\wedge : \boldsymbol{\omega} \in \mathbb{R}^3 \rightarrow so(3) \\ &[\boldsymbol{\omega}]^\wedge = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \end{aligned} \tag{10}$$

where $so(3)$ denotes the Lie algebra of $SO(3)$ which is the group of rotation matrices in the Euclidean space. The inverse operator of skew operator can be defined as follows:

$$\begin{aligned} &[.]^\vee : \boldsymbol{\Omega} \in so(3) \rightarrow \mathbb{R}^3 \\ &[\boldsymbol{\Omega}]^\vee \triangleq \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}^\vee \\ &= \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \end{aligned} \tag{11}$$

and the angular velocity can be calculated using the inverse operator:

$$\boldsymbol{\omega}_t \cong \frac{1}{T} \left[\Delta \boldsymbol{R}_e \, {\boldsymbol{R}_e}^T \right]^\vee \tag{12}$$

Note that the skew operator and its inverse are linear operators.

Let us define Δr as follows:

$$\dot{r}_t \triangleq \frac{\Delta r}{T} \tag{13}$$

As a result:

$$\Delta r \cong \begin{bmatrix} \Delta X_e \\ \left[\Delta \boldsymbol{R}_e \, {\boldsymbol{R}_e}^T \right]^\vee \end{bmatrix} \tag{14}$$

A. Case Study 1: Sampling Time as a Free Variable

By considering T as a free variable, the optimization problem (5) can be transformed into the following equivalent problem:

$$\begin{aligned} &\min_{\Delta \boldsymbol{q}, T} \Delta \boldsymbol{q}^T Q \Delta \boldsymbol{q} + \alpha T^2 \\ &\text{subject to} \\ &J \Delta \boldsymbol{q} = \Delta r \\ &b^- T \leq A \Delta \boldsymbol{q} \leq b^+ T \\ &\dot{\boldsymbol{q}}^- T \leq \Delta \boldsymbol{q} \leq \dot{\boldsymbol{q}}^+ T \\ &\epsilon \leq T \end{aligned} \tag{15}$$

where $0 < \epsilon \ll 1$, and $\alpha > 0$ are user-defined constants.

Let us introduce the following parameters:

$$\mathcal{X} = \begin{bmatrix} \Delta \boldsymbol{q} \\ T \end{bmatrix}, \quad Q_{\mathcal{X}} = \begin{bmatrix} Q & \boldsymbol{0}_{n \times 1} \\ \boldsymbol{0}_{1 \times n} & \alpha \end{bmatrix}, \quad \mathcal{J} = \begin{bmatrix} J & \boldsymbol{0}_{m \times 1} \end{bmatrix}$$

$$\mathcal{A} = \begin{bmatrix} A & -b^+ \\ -A & b^- \\ I_n & -\dot{\boldsymbol{q}}^+ \\ -I_n & \dot{\boldsymbol{q}}^- \\ \boldsymbol{0}_{1 \times n} & -1 \end{bmatrix}, \quad \mathcal{B} = \begin{bmatrix} \boldsymbol{0}_{n \times 1} \\ \boldsymbol{0}_{n \times 1} \\ \boldsymbol{0}_{n \times 1} \\ \boldsymbol{0}_{n \times 1} \\ -\epsilon \end{bmatrix} \tag{16}$$

where I_n and $\boldsymbol{0}_{k \times l}$ denotes, respectively, the identity matrix of dimension n and a zeros matrix of dimension $k \times l$. Recall that n and m are the dimension of $\Delta \boldsymbol{q}$ and Δr respectively.

Thus the optimization problem is transformed into the following classical QP problem:

$$\min_{\mathcal{X}} \mathcal{X}^T Q_{\mathcal{X}} \mathcal{X}$$

subject to

$$\begin{aligned} \mathcal{J}\mathcal{X} &= \Delta r \\ \mathcal{A}\,\mathcal{X} &\leq \mathcal{B} \end{aligned} \tag{17}$$

The above QP problem can be efficiently solved, in real-time, using an appropriate QP solver such as uQuadProg solver [9] or qpOASES solver [15].

Algorithm 1

1: Compute the jacobian matrix (J)
2: Compute Δr ▷ Eq. (14)
3: **procedure** INVERSE KINEMATICS($J, \Delta r, A, b^-, b^+, \dot{\boldsymbol{q}}^-, \dot{\boldsymbol{q}}^+$)
4: Solve the optimization problem (17)
5: **return** $\Delta \boldsymbol{q}$ and T
6: **go to** 1

B. Case Study 2: Incorporating End-effector Velocity Limits

By adding constraints on the end-effector velocity, the optimization problem (15) is transformed into the following optimization problem:

$$\min_{\Delta \boldsymbol{q}, T} \Delta \boldsymbol{q}^T Q \Delta \boldsymbol{q} + \alpha T^2$$

subject to

$$\begin{aligned} J\Delta \boldsymbol{q} &= \Delta r \\ b^-\, T \leq A\Delta \boldsymbol{q} &\leq b^+\, T \\ \dot{\boldsymbol{q}}^-\, T \leq \Delta \boldsymbol{q} &\leq \dot{\boldsymbol{q}}^+\, T \\ \epsilon &\leq T \\ \dot{r}^-\, T \leq \Delta r &\leq \dot{r}^+\, T \end{aligned} \tag{18}$$

where $\dot{r}^+$ and $\dot{r}^-$ are respectively the upper and lower limits of the end-effector velocity.

Note that the new constraint can be rewritten in an equivalent and more compact form as follows:

$$\dot{r}^-\, T \leq \Delta r \leq \dot{r}^+\, T \Leftrightarrow |\Delta r| \leq \dot{r}^+\, T \tag{19}$$

By introducing the following parameters:

$$\mathcal{X} = \begin{bmatrix} \Delta \boldsymbol{q} \\ T \end{bmatrix}, \quad Q_{\mathcal{X}} = \begin{bmatrix} Q & \mathbf{0}_{n\times 1} \\ \mathbf{0}_{1\times n} & \alpha \end{bmatrix}, \quad \mathcal{J} = \begin{bmatrix} J & \mathbf{0}_{m\times 1} \end{bmatrix}$$

$$\hat{\mathcal{A}} = \begin{bmatrix} A & -b^+ \\ -A & b^- \\ I_n & -\dot{\boldsymbol{q}}^+ \\ -I_n & \dot{\boldsymbol{q}}^- \\ \mathbf{0}_{1\times n} & -1 \\ \mathbf{0}_{m\times n} & -\dot{r}^+ \end{bmatrix}, \quad \hat{\mathcal{B}} = \begin{bmatrix} \mathbf{0}_{n\times 1} \\ \mathbf{0}_{n\times 1} \\ \mathbf{0}_{n\times 1} \\ \mathbf{0}_{n\times 1} \\ -\epsilon \\ -|\Delta r| \end{bmatrix} \tag{20}$$

Similarly to *Case Study 1*, the optimization problem is transformed into the following classical QP problem:

$$\min_{\mathcal{X}} \mathcal{X}^T Q_{\mathcal{X}} \mathcal{X}$$

subject to

$$\begin{aligned} \mathcal{J}\mathcal{X} &= \Delta r \\ \hat{\mathcal{A}}\,\mathcal{X} &\leq \hat{\mathcal{B}} \end{aligned} \tag{21}$$

Algorithm 2

1: Compute the jacobian matrix (J)
2: Compute Δr ▷ Eq. (14)
3: **procedure** INVERSE KINEMATICS($J, \Delta r, A, b^-, b^+, \dot{\boldsymbol{q}}^-, \dot{\boldsymbol{q}}^+, \dot{r}^+$)
4: Solve the optimization problem (21)
5: **return** $\Delta \boldsymbol{q}$ and T
6: **go to** 1

III. COMPLEXITY ANALYSIS

A complexity comparison with conventional inverse kinematics algorithms, Eq. (5), can give a clear idea about the possibility of an online implementation of the proposed algorithms.

1) **Algorithm 1**:
   - By comparing the optimization problems (5) and (17), the dimension of the optimization variable in the proposed algorithm ($\mathcal{X}$) is $n+1$ where n is the dimension of the conventional joints velocity variable ($\dot{\boldsymbol{q}}_t$).
   - One additional inequality constraint has been added.

   As a result, the computational complexity of **Algorithm 1** and of a conventional inverse kinematic algorithm are almost the same. However, **Algorithm 1** have a significant advantage of always providing a feasible solution.

2) **Algorithm 2**:
   - Similarly to **Algorithm 1**, the dimension of the optimization variable ($\mathcal{X}$) is $n+1$.
   - The number of inequality constraints has however been increased by $m+1$, where m is the dimension of the end-effector configuration vector.

   Consequently, the complexity of **Algorithm 2** and a conventional inverse kinematics problem are comparable. Recall that **Algorithm 2** can simultaneously handle constraints on the joint and the end-effector velocity limits, while conventional inverse kinematics algorithms simply cannot.

The numerical simulations in Section V also confirmed the above conclusions.

IV. PRACTICAL IMPLEMENTATION

Many of robotic systems have a control loop with a fixed time step. To take this constraint into account, the sampling time T should satisfies the following condition:

$$T = n\, T_s \tag{22}$$

where $n \in \mathbb{N}$ is a strictly positive integer, which becomes the new optimization variable, and T_s is the robot's control loop fixed time step.

Theoretically speaking, the optimization problems (17) and (21) become Mixed-Integer Quadratic Programming (MIQP) problems. However, in practice, solving a MIQP is generally more complex and computationally expensive than solving a standard QP.

In our special case, however, one can figure out that we do not need to solve MIQP problems, instead once T is obtained by solving (17) and (21), the parameter n can be easily obtained such as: $(n-1)\,T_s \leq T \leq n\,T_s$.

V. Simulation Results

We have conducted three simulations scenarios:

A. Scenario 1: Joint Velocity Limits

In order to validate the proposed algorithm for the case of joint velocity limits (**Algorithm 1**), we consider a planar redundant manipulator, which has 4 degrees of freedom (Fig. 1). The end-effector path is a Bezier curve defined by an initial, control and a goal positions. The joint velocity limits have been chosen as follows:

- $\dot{\boldsymbol{q}}^+ = -\dot{\boldsymbol{q}}^- = 0.5 \times [1 \quad 1 \quad 1 \quad 1]^T$

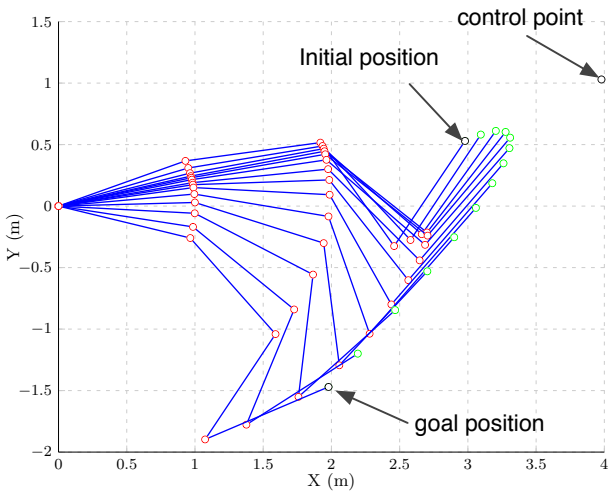


Fig. 1. Planar robot configurations

Fig. 2(a) points out that the constraints on the joint velocity are fully respected, and Fig. 2(c) shows that the error between the executed trajectory and the desired one is less than 10^{-5}m. To compare those results with a conventional inverse kinematics algorithm, the end-effector path has been first transformed into a trajectory by considering a uniform time distribution with a sampling time $T = 5 \times 10^{-3} s$ over the interval $[0, t_f]$, after several trials we found that the minimum t_f that makes the problem feasible is $t_f = 4.40\,s$. It is worth to mention that, at every trial, the conventional inverse kinematics algorithm stopped after few iterations because of an infeasibility issue. From Fig. 3 and Fig. 2, one can notice that:

- Our method automatically found that $t_f = 3.30\,s$ versus $t_f = 4.40\,s$ for the conventional method. This is because our method adapts the sampling time according to the joint velocity constraints; on the other hand, the conventional method requires a fixed sampling rate for the whole trajectory.
- Fig. 3(a) shows that the joint velocity constraints have been violated over the time interval $[0.02s,\ 0.83s]$, this is because the constraints of the optimization problem (5) cannot be simultaneously satisfied. The solver, in this case, tries first to satisfy the equality constraints and the inequality constraints are considered in a second priority. It is important to point out that the high oscillations in the joint velocity trajectories are not fully transmitted to the end-effector, this is because the robot is redundant with respect to the desired task.
- The end-effector tracking error in Fig. 3(c) is much bigger than that one of our method Fig. 2(c).

B. Scenario 2: Joint and End-effector Velocity Limits

To validate the proposed algorithm for the joint and end-effector velocity limits (**Algorithm 2**), we consider the same planar redundant manipulator in Scenario 1. The following parameters has been chosen:

- $\dot{\boldsymbol{q}}^+ = -\dot{\boldsymbol{q}}^- = 0.5 \times [1 \quad 1 \quad 1 \quad 1]^T$
- $\dot{r}^+ = -\dot{r}^- = [\dot{x}_t^+ \quad \dot{y}_t^+] = 0.7 \times [1 \quad 1]^T$

Fig. 4(a) and Fig. 4(b) show that the constraints on the joint and the end-effector velocity are fully respected, and Fig. 4(c) shows that the error between the executed trajectory and the desired one is less than 2.5×10^{-5}m. Note that the total time of the trajectory became $4.30\,s$, as it can be seen in Fig. 4(b), this is mainly to respect the end-effector velocity limits.

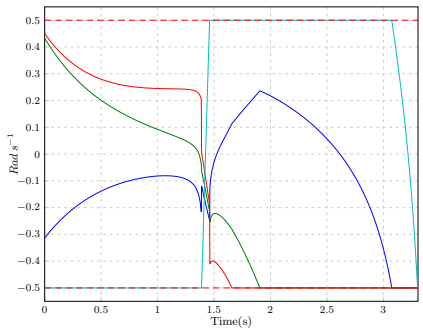
C. Scenario 3: Simulated Baxter Research Robot

A simulated scenario of a Baxter research robot and an obstacle that consists of a sphere attached to a thin cylindric rod is given in Fig. 5(a). The objective is to reach a goal pose (position and orientation) from an initial pose while avoiding the collision with the obstacles.

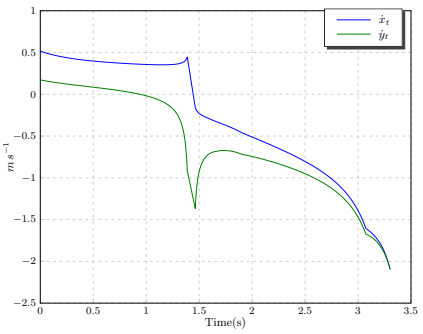
The following constraints have been considered:

1) Collision avoidance: the Baxter robot is approximated by its collision geometry model from the Unified Robot Description Format (URDF) file, where the arms are approximated by cylinders and boxes. The collision avoidance is formulated as inequality constraints [8] having the following form: $A\dot{\boldsymbol{q}}_t \leq b$.
2) Velocity limits on the robot's joints, these values are given in the URDF file.
3) Desired velocity limits (linear and angular velocities) of the right arm end-effector.

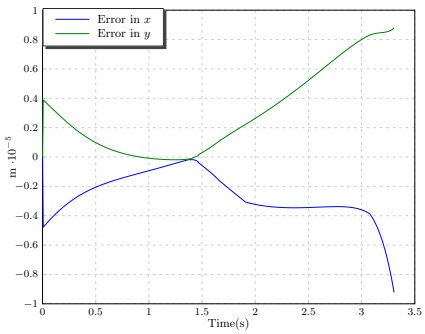
The formulation of the inverse kinematics problem in this scenario is however different, as the end-effector tries to reach the goal pose at each iteration via a direct straight line, at the same time the collision avoidance inequality constraints repulse the arm to keep a minimum clearance



(a) Joint velocities

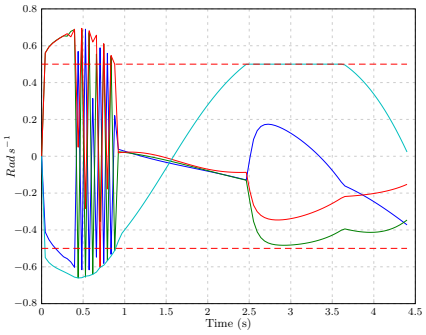


(b) End-effector velocity

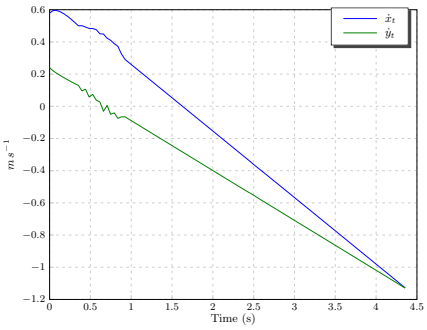


(c) End-effector tracking error

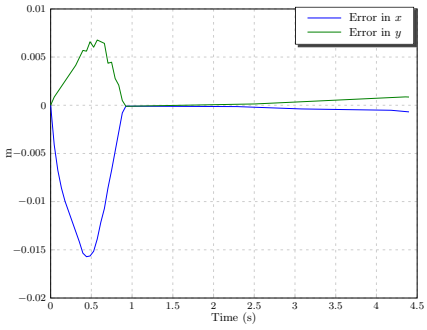
Fig. 2. **Algorithm 1**: joint velocity limits



(a) Joint velocities



(b) End-effector velocity



(c) End-effector tracking error

Fig. 3. Conventional inverse kinematics algorithm

to the obstacles. Snapshots of the simulated motion is given in Fig. 5(a). The right arm's joint trajectories are shown in Fig. 5(b).

VI. Conclusion and Future Work

In this paper, a new method for inverse kinematics while simultaneously considering joint and end-effector velocity limits is proposed. The method is simple to implement, yet efficient to handle velocity constraints. The proposed algorithms solve the problem of infeasibility that conventional inverse kinematics algorithms suffer from. These algorithms can be implemented online, and they have as input the geometric path of the end-effector that is converted, on-the-fly, to a feasible trajectory. Simulation results revealed the efficiency of the method to solve inverse kinematics problems in several scenarios. Future work will focus on the implementation of the proposed algorithms on a real robotic platform such as Baxter research robot or the humanoid robot HRP-2.

Acknowledgment

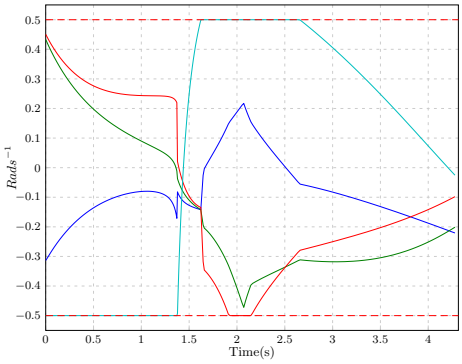
This research is partially supported by a discovery grant (Prof. Wael Suleiman) from the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

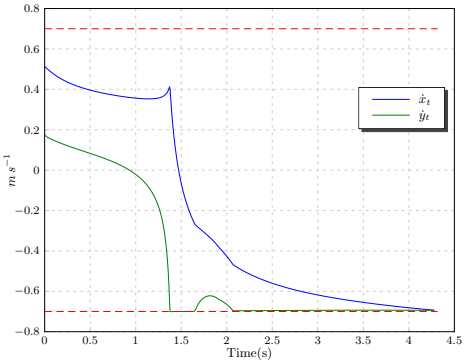
[1] M. Raghaven and B. Roth, "Kinematic analysis of the 6r manipulator of general geometry," in *The Fifth International Symposium on Robotics Research*, 1990, pp. 263–269.

[2] M. Raghavan and B. Roth, "Solving polynomial systems for the kinematic analysis and synthesis of mechanisms and robot manipulators," *Journal of Mechanical Design*, vol. 117, no. B, pp. 71–79, 06 1995.

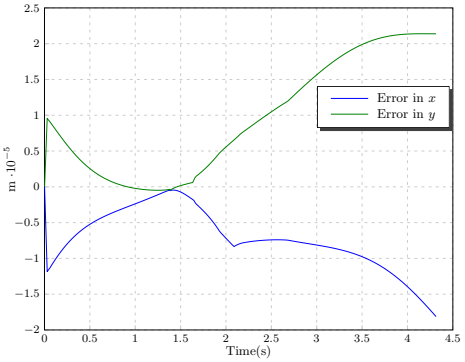
[3] Y. Nakamura, *Advanced Robotics: Redundancy and Optimization*. Addison-Wesley, 1991.



(a) Joint velocities

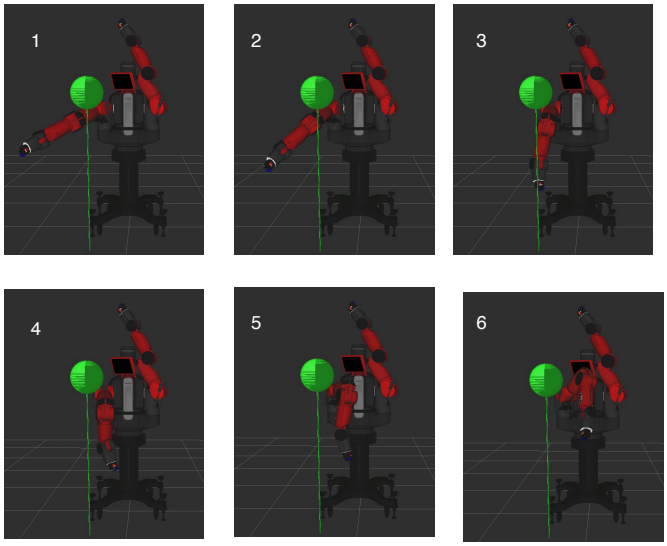


(b) End-effector velocity

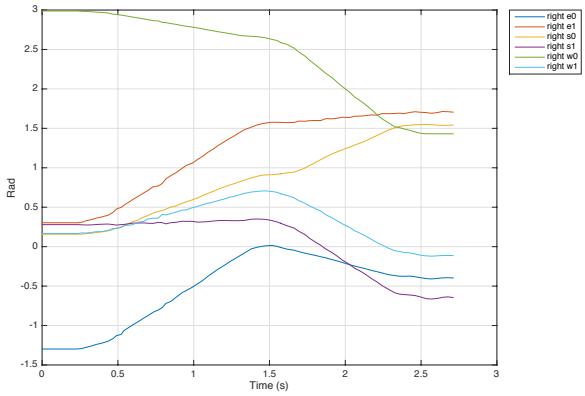


(c) End-effector tracking error

Fig. 4. **Algorithm 2**: joint and end-effector velocity limits



(a) Reaching scenario while avoiding collision



(b) Baxter right arm's joint trajectories

Fig. 5. Scenario 3: Simulated Baxter Research Robot

[4] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer, 2009.

[5] D. E. Whitney, "The mathematics of coordinated control of prosthetic arms and manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 94, no. 4, pp. 303–309, 12 1972.

[6] Y. Nakamura and K. Yamane, "Dynamics computation of structure-varying kinematic chains and its application to human figures," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 2, pp. 124–134, Apr 2000.

[7] S. Chan and P. Lawrence, "General inverse kinematics with the error damped pseudoinverse," in *IEEE International Conference on Robotics and Automation*, Apr 1988, pp. 834–839 vol.2.

[8] F. Kanehiro, F. Lamiraux, O. Kanoun, E. Yoshida, and J.-P. Laumond, "A Local Collision Avoidance Method for Non-strictly Convex Objects," in *2008 Robotics: Science and Systems Conference*, Zurich, Switzerland, June 2008.

[9] F. Kanehiro, M. Morisawa, W. Suleiman, K. Kaneko, and E. Yoshida, "Integrating geometric constraints into reactive leg motion generation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010, pp. 4069–4076.

[10] O. Kanoun, "Real-time prioritized kinematic control under inequality constraints for redundant manipulators," in *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011.

[11] A. Escande, N. Mansard, and P.-B. Wieber, "Fast resolution of hierarchized inverse kinematics with inequality constraints," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2010, pp. 3733–3738.

[12] ——, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *The International Journal of Robotics Research*, vol. 33, pp. 1006–1028, 2014.

[13] W. Suleiman, F. Kanehiro, E. Yoshida, J.-P. Laumond, and A. Monin, "Time parameterization of humanoid-robot paths," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 458 –468, June 2010.

[14] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*. Springer-Verlag, 2000, ch. Trajectory Planning, pp. 185–212.

[15] H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, 2014, (in print).