2018 IEEE International Conference on Robotics and Automation (ICRA)
May 21-25, 2018, Brisbane, Australia

A Novel Recurrent Neural Network for Improving Redundant Manipulator Motion Planning Completeness

Yangming Li[1], Shuai Li[2], and Blake Hannaford[3]

***Abstract*— Recurrent Neural Networks (RNNs) demonstrated advantages on control precision, system robustness and computational efficiency, and have been widely applied to redundant manipulator control optimization. Existing RNN control schemes locally optimize trajectories and are efficient and reliable on obstacle avoidance. However, for motion planning, they suffer from local minimum and do not have planning completeness. This work explained the cause of the planning incompleteness and addressed the problem with a novel RNN control scheme. The paper presented the proposed method in detail and analyzed the global stability and the planning completeness in theory. The proposed method was compared with other three control schemes on the precision, the robustness and the planning completeness in software simulation and the results shows the proposed method has improved precision and robustness, and planning completeness.**

***Index Terms*— Motion Planning, Kinematic Control, Recurrent Neural Networks, Redundant Manipulator, Robot**

I. Introduction

Manipulator motion planning is a process of finding a valid sequence of control commands that moves the manipulator end effector from the initial position to the desired goal without breaking the constraints or collision. As fundamental as motion planning is, it has been proved PSPACE-hard and remains a challenging problem for redundant manipulators[1].

Motion planning is widely studied in robotics. For mobile robots, heuristic search algorithms, such as A^*, demonstrated high performance and accuracy, and are widely adopted[2]. However, those algorithms are typically inefficient for manipulators, because examination of heuristic results in the configuration space is computationally prohibitive under the constraints of manipulators[3]. For manipulators with redundancy, the planning is even more complicated, thus sampling based algorithms are often adopted to "approximate" the solution without considering all constraints[4]. As a result, planned results might be infeasible to robots[3], [4]. Aiming to address this problem, another category of algorithms is based on constrained optimization and uses powerful mathematical tools such as covariant Hamiltonian optimization[3], derivative-free stochastic optimization[5] etc., to achieve fast convergence and high success rate. However, those methods may converge to local minima, and lack planning completeness [6].

Recurrent Neural Networks (RNNs) based algorithms have been broadly applied improving the computational efficiency and the robustness of mobile robot localization[7], robot arm collaborative control[8], natural language processing[9], and addressing environmental dynamicity[10]. In the domain of redundant manipulator motion planning, Xia *et al.* proposed a RNN control scheme to optimize joint velocities for trajectory tacking with serial redundant manipulators[11]. Zhang *et al.* proposed a RNN control scheme for optimizing the motion in order to minimize the energy consumption[12]. Li *et al.* optimized manipulators collaborative motions in distributed systems with RNNs[13]. Zhang *et al.* extended RNNs to obstacle avoidance by converting the collision avoidance condition into a constraint for RNN neural activities[14]. Guo *et al.* proposed to optimize manipulator joint accelerations for obstacle avoidance and velocity smoothness improvement[15]. More discussions on RNNs based control schemes can be found in[16].

Despite of the advantages of existing RNN control schemes, these algorithms are by nature a constrained optimization algorithm thus suffer from the local minimum problem. This is because these RNN control schemes are globally attracted by the target but only optimize locally. This paper proposes a novel RNN control scheme that has probabilistic planning completeness through globally exploring the workspace. The proposed method inherits the control precision, the robustness and the efficiency from RNNs and most importantly, the planning results are guaranteed executable. In summary, the main contributions of this work are:

- We propose a novel RNN control scheme that has planning completeness and does not suffer from the local minimum problem. It also has global stability, no error accumulation and improved control precision.
- We prove the global stability and the planning completeness in theory.
- We demonstrate the application of the proposed method and compared it with three other control schemes in terms of control precision, robustness against noise and planning completeness.

II. Redundant Manipulator Motion Planning with Random RNN

A. Kinematic Control for Redundant Manipulator

Manipulator kinematic model defines the nonlinear mapping from the end effector pose (in the task space) to the joint states (in the configuration space) as $\boldsymbol{r}(t) = f\big(\boldsymbol{q}(t)\big)$,

[1]Department of Electrical Engineering, University of Washington, Seattle, WA, USA 98195

[2]Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong

[3]Departments of Electrical Engineering, Bioengineering, Mechanical Engineering, and Surgery, University of Washington, Seattle, WA, USA 98195

where $\boldsymbol{q}(t) \in \mathbb{R}^{\mathrm{m}}$ is the joint state vector and $\boldsymbol{r}(t) \in \mathbb{R}^{\mathrm{n}}$ is the end effector pose vector, $f(\cdot)$ is the kinematic model. The kinematic control problem is to find the corresponding $\boldsymbol{q}(t)$ for a given $\boldsymbol{r}(t)$, as $\boldsymbol{q}(t) = f^{-1}\big(\boldsymbol{r}(t)\big)$.

For almost all kinematic redundant manipulators, the mapping $f(\cdot)$ are nonlinear and non-convex. The mapping can be projected into the velocity space through differentiating with respect to time, as: $\dot{\boldsymbol{r}}_{\mathrm{t}} = J\dot{\boldsymbol{q}}_{\mathrm{t}}$, where J is the $n \times m$ Jacobian matrix. For non-redundant manipulators, the joint states are fully defined by the task and $m = n$, if J is full rank, we have $\dot{\boldsymbol{q}}_{\mathrm{t}} = J^{-1}\dot{\boldsymbol{r}}_{\mathrm{t}}$, which means for a given $\dot{\boldsymbol{q}}_{\mathrm{t}}$, the corresponding $\dot{\boldsymbol{r}}_{\mathrm{t}}$ is uniquely determined. For redundant manipulators, because $m > n$, there exists infinite amount of $\dot{\boldsymbol{q}}_{\mathrm{t}}$ that correspond to the desired $\dot{\boldsymbol{r}}_{\mathrm{t}}$. Those $\dot{\boldsymbol{q}}_0$ that correspond to self-motions forms the null-space of the Jacobian as $J\dot{\boldsymbol{q}}_0 = \boldsymbol{0}$. While the existence of the null space leads to great versatility and broad applicability, how to choose the optimal one is an interesting and attractive question.

One family of the classical solutions that choose the optimal one is named "Gradient Projection Method", because those algorithms utilize the gradient of a defined cost function $h(\boldsymbol{q})$ to determine a joint velocity vector, $\dot{\boldsymbol{q}}_0$[17], [18]. The components of $\dot{\boldsymbol{q}}_0$ that locate in the null space of J can be selected by $(I - J^{\dagger}J)$. By adding the selected components to the motion of moving end-effector($J^{\dagger}\dot{\boldsymbol{r}}$), the optimal joint velocity will be achieved as: $\dot{\boldsymbol{q}} = J^{\dagger}\dot{\boldsymbol{r}} + (I - J^{\dagger}J)\dot{\boldsymbol{q}}_0$, where $J^{\dagger}$ is the Moore-Penrose pseudo-inverse defined as $J^{\dagger} = J^{\mathrm{T}}(JJ^{\mathrm{T}})^{-1}$.

Another family of the classical solutions is named as task space augmentation algorithms. In contrast to the Gradient Projection family algorithms, these algorithms solve the redundancy by "augmenting" the Jacobian as J_{A}=$[J^{\mathrm{T}}, J_{\mathrm{y}}^{\mathrm{T}}]^{\mathrm{T}}$, and the corresponding model changes to $\dot{\boldsymbol{r}}_{\mathrm{At}} = J_{\mathrm{A}}\dot{\boldsymbol{q}}_{\mathrm{t}}$, where $\boldsymbol{r}_{\mathrm{A}}(t) = [f(\boldsymbol{q}(t))^{\mathrm{T}}, f_y(\boldsymbol{q}(t))^{\mathrm{T}}]^{\mathrm{T}}$, $f_{\mathrm{y}}(\boldsymbol{q}(t)) \in \mathbb{R}^{\mathrm{p}}$ is a functional constraint task, where $p \leq (m-n)$[17], [19]. There are many ways for defining the functional constraint task. If we define the task by projecting the gradient of a defined cost function, $h(\boldsymbol{q})$, onto the null space of the Jacobian and impose $[I - JJ^{\dagger}](\frac{\partial h(\boldsymbol{q})}{\partial \boldsymbol{q}})^{\mathrm{T}} = 0$, where $p = m - n$, the method will produce the same results as the equivalent Gradient Projection method does.

B. RNN Motion Planning and Precision and Robustness Improvement

RNN control schemes avoid the inversion of the Jacobian matrices and effectively improved the efficiency and the robustness of redundant manipulator control. And how to solve the redundant manipulator control problem with RNN has been mathematically explained in [16]. In order to apply RNN for motion planning, the following problems need to be addressed: 1) the control precision, 2) the local minima in exploration, 3) the exploration efficiency.

The control precision is especially important in motion planning because collisions are fatal failures. In real world applications, manipulators stay away from obstacles with a safety distance, and this threshold is often derived from the control precision. While it is trivial to prove that larger threshold decreases the manipulator workspace, high control precision is desired for improving applicability.

It has been proven that the error of the control scheme [11]

$$\varepsilon\dot{\boldsymbol{u}} = -\boldsymbol{u} + P_{\Omega}(\boldsymbol{u} - \frac{\partial L}{\partial \boldsymbol{u}}) \tag{1a}$$

$$\varepsilon\dot{\boldsymbol{\lambda}} = \dot{\boldsymbol{r}}_{\mathrm{d}} - Ju, \tag{1b}$$

converges to zeros:

$$\dot{\boldsymbol{r}} - \dot{\boldsymbol{r}}_{\mathrm{d}} = \dot{\boldsymbol{r}} - J\dot{\boldsymbol{q}}, \tag{2}$$

and RNN control schemes are more precise than comparable numerical methods[20]. However, we did find this control scheme suffers from error accumulation, as explained below.

From Eqn.1b we have[21]:

$$\begin{aligned} \lambda =& \lambda_0 + \frac{1}{\varepsilon}\int(\dot{\boldsymbol{r}}_{\mathrm{d}} - Ju)\,dt \\ =& \lambda_0 + \frac{1}{\varepsilon}(\boldsymbol{r}_{\mathrm{d}} - \boldsymbol{r}_{d0} - \int Ju\,dt), \end{aligned} \tag{3}$$

where λ_0 and $\boldsymbol{r}_{d0}$ denotes the value of λ and $\boldsymbol{r}_d$ at $t = 0$, respectively.

By replacing λ in Eqn.1 with Eqn.3 we have:

$$\begin{aligned} \varepsilon\dot{\boldsymbol{u}} =& -\boldsymbol{u} + P_{\Omega}\Big(J^{\mathrm{T}}\big(\lambda_0 + \frac{1}{\varepsilon}(\boldsymbol{r}_{\mathrm{d}} - \boldsymbol{r}_{d0} - \int Ju\,dt\big)\Big) \\ =& -\boldsymbol{u} + P_{\Omega}\Big(J^{\mathrm{T}}\big(\lambda_0 + \frac{1}{\varepsilon}(\boldsymbol{r}_{\mathrm{d}} - \boldsymbol{r}_{\mathrm{d}0} - (\boldsymbol{r} - \boldsymbol{r}_0)\big)\Big). \end{aligned} \tag{4}$$

From Eqn. 2 and 4 we know that given arbitrary time point $t = 0$, as long as error $\boldsymbol{e}_0 = \boldsymbol{r}_{d0} - \boldsymbol{r}_0 \neq 0$, the error will be accumulated with time.

Under the RNN architecture, this problem can be addressed by feeding the error, $\boldsymbol{e} = \boldsymbol{r}_{\mathrm{d}} - \boldsymbol{r}$, back into the neural network as[21]:

$$\min_{\boldsymbol{u}}(\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u} + k\boldsymbol{e}^{\mathrm{T}}\boldsymbol{e}), \tag{5}$$

where $k > 0$ is a weighting factor.

In order to comply with the constraints such as the joint limits, the proposed method adopted the projection function P_{Ω} to bound the neural activities, which is defined as:

$$P_{\Omega}(x) = \begin{cases} d^- & \text{for} \quad x \leq d^- \\ x & \text{for} \quad d^- < x < d^+ \\ d^+ & \text{for} \quad d^+ \leq x, \end{cases} \tag{6}$$

with boundary conditions as:

$$\begin{cases} d^- & = \max\left(-c_1(\boldsymbol{q} - \boldsymbol{q}^-), w^-\right) \\ d^+ & = \min\left(-c_2(\boldsymbol{q} - \boldsymbol{q}^+), w^+\right), \end{cases} \tag{7}$$

where $\boldsymbol{q}$ denotes the joint angle, $\boldsymbol{q}^+$ and $\boldsymbol{q}^-$ are the upper and the lower bounds of joint limits (from manipulator mechanics and application requirements); w^+ and w^- are the upper and the lower bounds of the joint speeds, and c_1 and c_2 are two positive scaling factors. The boundary conditions dynamically change with joint states and are also absolutely bounded by fixed the thresholds. The advantages of this design are it ensures the solutions meeting the joint (physical) limits, and it also avoids the infinite increase or decrease of joints' speeds.

Therefore, the RNN control scheme described by Eqn.5, 6 and 7 has improved control precision and stability. The corresponding neural dynamics is:

$$\boldsymbol{u} = P_{\Omega}(-kJ^{\mathrm{T}}(\boldsymbol{r} - \boldsymbol{r}_{\mathrm{d}})) \tag{8a}$$

$$\dot{\boldsymbol{r}}_{\mathrm{d}} = Ju. \tag{8b}$$

C. Obstacle Avoidance in RNN Motion Planning

Potential collisions can be detected by measuring distance between a robot arm and obstacles. When potential collisions exist, there are multiple ways to avoid collision with RNN control schemes. One way is to convert the collision condition into a constraint and to augment the task space, and to optimize the motion with respect to the augmented task space. The second way is to convert the collision condition into the neural activity boundary conditions. Both of the two ways are constrained local optimization. There is also the third way to avoid collision, which is simply stopping the manipulator. The third way is computationally efficient as it avoids to locally optimizing motion, but it leads to decreased workspace.

In the proposed method, we adopt the second way if the manipulator moves toward the real target, and adopt the third way if the manipulator moves toward a random target. This is to ensure the planning completeness while maintaining efficiency at the same time.

The third way is self-explained, and also is easy to implement. We just need to ask the RNN control scheme to output zeros if the potential collision has been detected and to report failure for the planner. For the second way, intuitively, the potential collision can be avoided if the manipulator *does not* move any closer to obstacles. In the velocity space, it can be implemented by exert an "escaping" velocity to the potential collision point, $\boldsymbol{r}_{\mathrm{o}}$, on the manipulator. If we denote the potential collision point on the obstacle as: o_{o}, the direction of this escaping velocity can be determined as:

$$\dot{\boldsymbol{r}}_{\mathrm{o}} = [r_{ox} - o_{ox}, r_{oy} - o_{oy}, r_{oz} - o_{oz}]^{\mathrm{T}}, \tag{9}$$

where the subscript x, y and z denote the 3-dimensional coordinates. Therefore, the collision can be avoided if the velocity of the potential collision link is in the range of zero and $b\dot{\boldsymbol{r}}_{\mathrm{o}}$, where b is a non-negative scaling factor.

Notice this velocity range is in the task space, and it can be converted into the configuration space as: $J_{\mathrm{o}}\dot{\boldsymbol{q}} = \dot{\boldsymbol{r}}_{\mathrm{o}}$, and the output of the neural network (d_o^-, d_o^+) corresponds to the velocity limit in the configuration space. Therefore, the boundary conditions described in Eqn.7 become:

$$(d'^-, d'^+) = (d^-, d^+) \cup (d_o^-, d_o^+). \tag{10}$$

D. Motion Planning Completeness and Efficiency

We explained in the introduction section that classical RNN control schemes suffer from planning incompleteness, because they modeled the planning problem as constrained optimization. In the proposed method, we use randomness to address the planning incompleteness. Actually randomness has been widely used in the robotic community to address various high-dimensional and non-convex optimization problems. For example, random sampling has been used to address the data association problem, if the depth of the hypothesis tree is big[22], [23]. From the Neuroscience and the machine learning perspective, randomness is also widely used for improving efficiency, because it is considered as being correlated with superior learning ability[24]. For example, through utilizing randomness, Bayesian Program Learning demonstrated the ability of Human-level concept learning[24].

In the proposed method, we address the planning incompleteness problem through randomizing the network inspiration, which can be mathematically explained as:

$$\dot{\boldsymbol{r}}_{\mathrm{d}} = g \cdot (\boldsymbol{r}_{\mathrm{random}} - \boldsymbol{r}), \tag{11}$$

where $\boldsymbol{r}_{random} \in \mathbb{R}^{\mathrm{n}}$ is a random goal in the task space, and g is a non-zero weight that regulates the tracking velocity.

Intuitively, the proposed scheme will be attracted by random goals; therefore the full configuration space can be explored, given sufficiently long time. However, this scheme is not guaranteed to be better than brute force search and might be impractical to use. This is because the classical theory of probability ensures that the random goal will almost surely be reached in infinite time with random sampling, because it samples a point in a space. This time can be shortened if we are satisfied with reaching the neighborhood of the goal. If the neighborhood is defined as a sphere with radius r, the probability of sampling the goal is approximately $\frac{4\pi r^3}{3V}$, where V is the volume of the task space. Although increasing r leads to fast convergence, the planning results may be not realistic, due to the existence of robot arm joint limits and obstacles.

A practical solution will be performing both the random exploration and the directional search, as explained in Algorithm 1. From Algorithm 1 we know that the random exploration will explore the entire task space. In this process, the generated movement trajectory is guaranteed executable, because the proposed RNN control scheme considered both the obstacle avoidance and the joint limits. The directional search directly drives the end effector towards the goal position, starting from the a point that has been previously reached in the random exploration. In this process, the proposed scheme performs local optimization to search "corners" in the configuration space.

In Algorithm 1, N_{Loops} denotes the maximum allowed exploration; p_{s} is a scalar between 0 and 1, which balances the random exploration and the directional search. Because both the random exploration and the directional search utilize the same RNN control scheme, all constraints, such as the joint limits and obstacle avoidance are ensured. Meanwhile, the random exploration ensures the proposed RNN will not be trapped by local minima.

III. Theoretical Analyses

A. Stability

The projection function defined in Eqn.6 ensures: $d'^- \leq \boldsymbol{x} \leq d'^+$. Furthermore, from Eqn.7 it is clear that the neural

Algorithm 1 Complete Motion Planning with RNN

Input: Target ($\boldsymbol{r}_g$), Manipulator Start Position ($\boldsymbol{r}_0$)
Output: Sequence of Joint States (C)
Init : $\boldsymbol{r}_s$=$\boldsymbol{r}_0$, $\boldsymbol{r}_d$=$\boldsymbol{r}_0$, $T_s = \{[\boldsymbol{r}_s, \boldsymbol{r}_d, \emptyset]\}$, RNN, N_{Loops}
if rand(0,1)$> p_{\text{s}}$ **then**
  $\boldsymbol{r}_{\text{d}} =$ A random point in the task space
else
  $\boldsymbol{r}_{\text{d}} = \boldsymbol{r}_g$
end if
Select the reached goals from T_s, which is closest to $\boldsymbol{r}_{\text{d}}$, use it as $\boldsymbol{r}_s$
Produce a sequence of commands $c_{s,d}$ with the control scheme defined in Eqn 8 and 10, for given $\boldsymbol{r}_s$ and $\boldsymbol{r}_{\text{d}}$
while N_{Loops}>0 **do**
  if ($\boldsymbol{r}_{\text{d}}$ is reached) **then**
    if ($\boldsymbol{r}_{\text{d}} == \boldsymbol{r}_g$) **then**
      Back tracing control sequence $C = \cup c_{s,d}$.
      return C
    end if
  else
    $\boldsymbol{r}_{\text{d}} =$ the true end effector position.
  end if
  Insert $[\boldsymbol{r}_s, \boldsymbol{r}_d, c_{s,d}]$ into T_s
  N_{Loops} = N_{Loops}-1
end while
return $\emptyset$

activities are strictly bounded in the range of (w^-, w^+). In order to prove the stability, let's define Lyapunov function as $V = \boldsymbol{e}^{\text{T}}\boldsymbol{e}/2$, where $\boldsymbol{e} = \boldsymbol{r}_{\text{d}} - \boldsymbol{r}$ is the penalty from tracking error (defined in Eqn. 5). Therefore, it can be proved (Eqn. 20$\sim$37 in [25]) that the largest invariant set for $\dot{v} = 0$ only contains $\boldsymbol{e} = 0$, which indicates the global stability[11], [12].

B. Probability Completeness

If we denote the task space of a manipulator as $X \subset \mathbb{R}^{\text{n}}$ and the obstacle space as: $X_{\text{obs}} \subset X$, the reachable subspace becomes $X_{\text{reach}} \subset X_{\text{free}}$, where X_{free} is the free space as: $X_{\text{free}} = X \backslash X_{\text{obs}}$. And the probabilistic planning completeness is defined as: for a given target $x_{\text{d}} \in X_{\text{reach}}$, if the set of valid paths, $\Sigma_\delta = \{\delta : [0,t] \to x_\delta\}$ equals to $\emptyset$, it is reported in finite time, otherwise $P(\Sigma_\delta \cap T_{\text{s}} = \emptyset) = 0$, where T_{s} denotes the set that contains all valid paths.

For all points in the reachable space ($\forall x_{\text{d}} \in X_{\text{reach}}$), there exists a space, B_{d} for x_{d}, and for all points in the space ($\forall x_{\text{j}} \in B_{\text{d}}$), a valid path can be found by the control scheme described in Eqn. 8. Intuitively, B_{d} can be imagined as a basin to x_{d}. And because all points in the reachable space are reachable, $\cup B_{\text{d}} \geq X_{\text{reach}}$ holds truth. Therefore, with the proposed random RNN, the probability of reaching each of B_{d} is non-zero, therefore, a valid path can be found, as long as it exists.

IV. Illustrative Examples and Discussion

It has been proven that RNN control schemes have advantages on computational efficiency, precision and robustness[20]. In this section, we compared the proposed with other RNN control schemes, to reveal the domains that the proposed method is best suited for.

Table. I compares RNN control schemes for redundant manipulator motion planning. To our best knowledge, the proposed method is the first RNN control scheme that achieved planning completeness. Among algorithms listed in the table, three were chosen based on the similarities to the proposed method. The first algorithm is from [14]. This RNN control scheme addressed the obstacle avoidance problem for redundant manipulators in the velocity space, and we denoted it as "Method1" in this paper. The second algorithm[15] is also capable of obstacle avoidance, but optimizes in the acceleration space, and we denoted it as "Method2". The third method[25] optimizes motion in obstacle-free environments, but it addressed the error accumulation problem and achieved high control precision and robustness, thus we also compared with it and denoted it as "Method3".

We compared the four algorithms with Mitsubishi PA10-7C based simulation. The PA10 redundant manipulator has 7 DoF and its mechanic structure is similar to human arms (Table. II). Moreover, its kinematic simulator has been well studied[28]. In the simulation experiments, the parameters of the proposed method were empirically chosen as: $p_{\text{s}} = 0.5$, $k = 100$ and $c_1 = c_2 = 0.5$. For the other methods, the parameters were chosen according to the corresponding references[14], [15], [21].

A. Control Precision and Robustness against Noise

The proposed method is not only a motion planner, but also a control scheme. We first studied the control precision and the robustness against noise with simulation experiments.

Because Method3 is not capable of obstacle avoidance and avoiding obstacle does not impact the control precision (but slows down the convergence), the four algorithms are compared in obstacle-free environments for the precision and the robustness comparison. From Fig.1 we can see that all four algorithms are competent on the point target tracking task.

Gaussian White Noise with different level of standard deviation: $\sigma = 0.01$, $\sigma = 0.05$ and $\sigma = 0.25$, were injected into the neural networks' outputs as additive noise, in order to test the robustness against process noise. The averaged tracking errors from the four algorithms are compared in Table III. From the table we can see that the proposed method has strong robustness against noise due to the fact that it closes the loop of with tracking errors.

B. Planning Completeness

Because "Method3" is not capable of obstacle avoidance, in this subsection, only "Method1" and "Method2" were compared with the proposed method.

Two scenarios that are popular in manipulator control applications are used in simulation experiments. Scenario 1 (visually explained in Fig. 2) is environments with a plane-like obstacle and Scenario 2 (Fig.3) is environments with a window-shaped obstacle. In both scenarios, the PA10

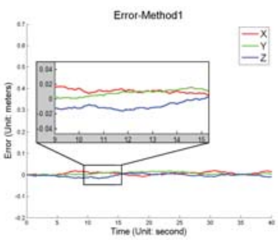
TABLE I: Comparisons among Recurrent Neural Network based Control Schemes for Redundant Manipulators†.

	Global Convergence	Theoretical Error	Free of Error Accumulation	Free of Matrix Inversion	Physical Limits Avoidance	Smothness Optimization	Obstacle Avoidance	Planning Completeness
Model in [26]	Yes	Nonzero	No	No	No	No	No	No
Model in [27]	Yes	Zero	No	Yes	Yes	No	No	No
Model in [21]	Yes	Zero	Yes	Yes	Yes	Yes	No	No
Model in [20]	Yes	Zero	Yes	Yes	Yes	Yes	No	No
Model in [14]	Yes	Zero	No	Yes	Yes	No	Yes	No
Model in [15]	Yes	Zero	No	Yes	Yes	Yes	Yes	No
Proposed	Yes	Zero	Yes	Yes	Yes	Yes	Yes	Yes‡

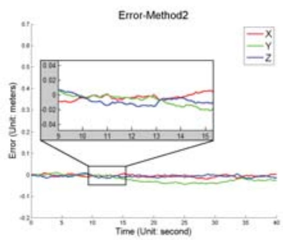
†Algorithms that are most closely related to the proposed method are selected, and a thorough review can be found in [16] .
‡Probabilistic completeness.

TABLE II: Denavit-Hartenberg Parameters of Mitsubishi PA10-7C 7 DoF Redundant Manipulator.

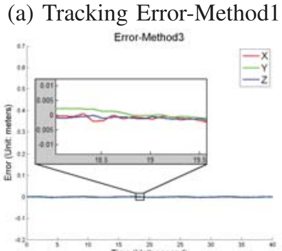
Link(i)	α_{i-1}	a_{i-1}	d_i	θ_i
1	$-\pi/2$	0	317mm	θ_1
2	$\pi/2$	0	0	θ_2
3	$-\pi/2$	0	450mm	θ_3
4	$\pi/2$	0	0	θ_4
5	$-\pi/2$	0	450mm	θ_5
6	$\pi/2$	0	0	θ_6
7	0	0	70mm	θ_7



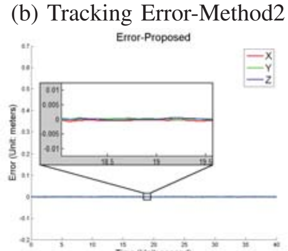
(a) Tracking Error-Method1



(b) Tracking Error-Method2



(c) Tracking Error-Method3



(d) Tracking Error-Proposed

Fig. 1: Precision Comparison in Point Target Tracking Task.

manipulator started from random initial positions and under the control of Method1, Method2 and the proposed method, respectively. In Fig. 2 and 3, semi-transparent blue planes denote obstacles; thick colored lines indicate manipulator initial configurations and the thin ones are the trajectory. Goal is denoted by a red sphere, and the curved red line segments denote the end effector trajectories.

The success rates are listed in Table IV. The success rate is defined as: v_s/v_e, where v_s denotes the total number of experiments, in which the manipulator reached the target, and $v_e = 50$ denotes the total number of experiments. The two scenarios with example planning results are visualized in Fig.2 and 3. From the figures and the table we can see that because of the local minimum problem, only the proposed method succeed in motion planning in all simulation experiments.

TABLE III: RMS Position Tracking Error With Respect to Various Noise Level.

	$\sigma = 0.01$	$\sigma = 0.05$	$\sigma = 0.25$
Proposed	0.007	0.010	0.010
Scheme1[14]	0.027	0.092	0.246
Scheme2[15]	0.025	0.069	0.199
Scheme3[25]	0.005	0.010	0.011

TABLE IV: Planning Success Rate Comparison in the Two Scenarios.

	Method1	Method2	Proposed
Scenario 1	92%	92%	100%
Scenario 2	42%	44%	100%

V. Conclusion

RNN control schemes avoid the inversion of the Jacobian matrix and demonstrated improved control precision, robustness and efficiency. Although these algorithms have attracted broad attention and have been extensively applied to redundant manipulators recently, they did not yet have planning completeness.

This work presented a novel Recurrent Neural Network (RNNs) based control scheme to address the planning incompleteness problem. The proposed method not only has the planning probabilistic completeness, but also ensures the planning result is executable in practice. Meanwhile, the proposed method inherits high control precision and robustness from RNNs. The proposed method is designed for addressing the single-query path planning problem in unknown environments. The planning results can be further optimized given partial or full knowledge of environmental information, which is also the next step of this work.

Acknowledgment

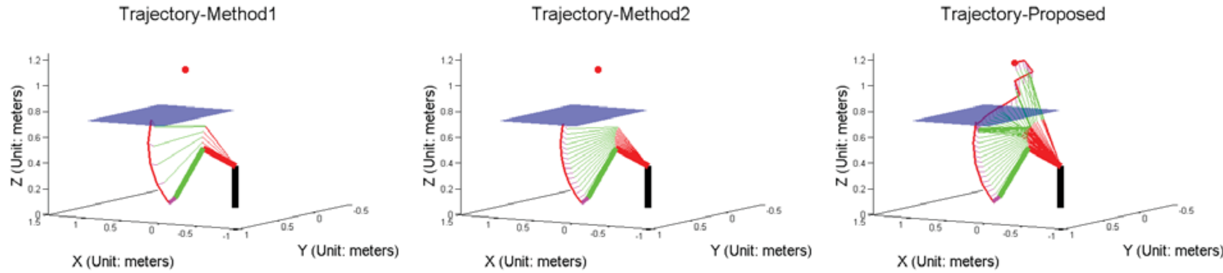
This work was supported by NSF grant IIS-1637444, NIH grant 5R21EB016122-02 and the Korean Institute of Science and Technology (KIST); also, the authors would like to thank the reviewers.

References

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

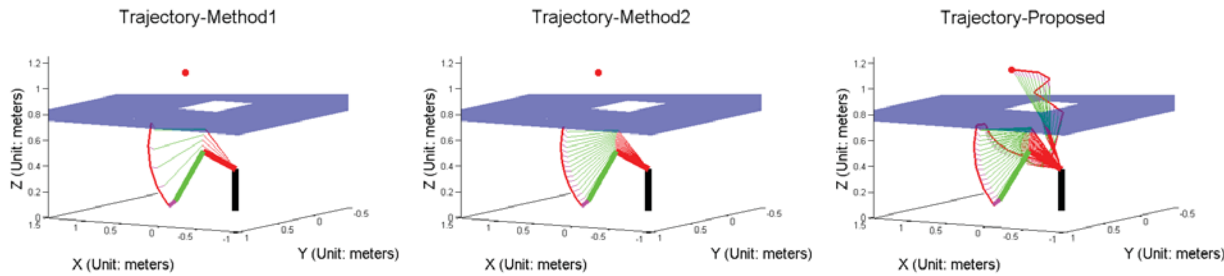
[2] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[3] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.



(a) Method1 (b) Method2 (c) Proposed

Fig. 2: Example Planning Results in an Environment with a Plane-shaped Obstacle.



(a) Method1 (b) Method2 (c) Proposed

Fig. 3: Example Planning Results in an Environment with a Window-shaped Obstacle.

[4] M. V. Weghe, D. Ferguson, and S. S. Srinivasa, "Randomized path planning for redundant manipulators without inverse kinematics," in *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*. IEEE, 2007, pp. 477–482.

[5] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 4569–4574.

[6] J.-J. Kim and J.-J. Lee, "Trajectory optimization with particle swarm optimization for manipulator motion planning," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 620–631, 2015.

[7] Y. Li, J. Zhang, and S. Li, "STMVO: biologically inspired monocular visual odometry," *Neural Computing and Applications*, pp. 1–11, 2016.

[8] L. Jin, S. Li, X. Luo, Y. Li, and B. Qin, "Neural dynamics for cooperative control of redundant robot manipulators," *IEEE Transactions on Industrial Informatics*, 2018.

[9] T. Mikolov, S. Kombrink, L. Burget, J. Černockỳ, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5528–5531.

[10] Y. Li, S. Li, and Y. Ge, "A biologically inspired solution to simultaneous localization and consistent mapping in dynamic environments," *Neurocomputing*, vol. 104, pp. 170–179, 2013.

[11] Y. Xia and J. Wang, "A dual neural network for kinematic control of redundant robot manipulators," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 31, no. 1, pp. 147–154, 2001.

[12] Y. Zhang, J. Wang, and Y. Xia, "A dual neural network for redundancy resolution of kinematically redundant manipulators subject to joint limits and joint velocity limits," *IEEE transactions on neural networks*, vol. 14, no. 3, pp. 658–667, 2003.

[13] S. Li, J. He, Y. Li, and M. U. Rafique, "Distributed recurrent neural networks for cooperative control of manipulators: A game-theoretic perspective," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 2, pp. 415–426, 2017.

[14] Y. Zhang and J. Wang, "Obstacle avoidance for kinematically redundant manipulators using a dual neural network," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 1, pp. 752–759, 2004.

[15] D. Guo and Y. Zhang, "Acceleration-level inequality-based man scheme for obstacle avoidance of redundant robot manipulators," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 12, pp. 6903–6914, 2014.

[16] Y. Zhang and Z. Zhang, *Repetitive motion planning and control of redundant robot manipulators*. Springer Science & Business Media, 2014.

[17] B. Siciliano, "Kinematic control of redundant robot manipulators: A tutorial," *Journal of Intelligent & Robotic Systems*, vol. 3, no. 3, pp. 201–212, 1990.

[18] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: a survey," *Cognitive processing*, vol. 12, no. 4, pp. 319–340, 2011.

[19] A. M. Zanchettin and P. Rocco, "A general user-oriented framework for holonomic redundancy resolution in robotic manipulators using task augmentation," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 514–521, 2012.

[20] Z. Zhang, L. Zheng, J. Yu, Y. Li, and Z. Yu, "Three recurrent neural networks and three numerical methods for solving repetitive motion planning scheme of redundant robot manipulators," *IEEE/ASME Transactions on Mechatronics*, 2017.

[21] Y. Li, S. Li, D. Caballero, M. Miyasaka, A. Lewis, and B. Hannaford, "Improving control precision and motion adaptiveness for surgical robot with recurrent neural network," in *Proceedings of the... IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.

[22] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, June 1981.

[23] Y. Li, S. Li, Q. Song, H. Liu, and M. Q.-H. Meng, "Fast and robust data association using posterior based approximate joint compatibility test," *Industrial Informatics, IEEE Transactions on*, vol. 10, no. 1, pp. 331–339, 2014.

[24] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[25] S. Li, Y. Zhang, and L. Jin, "Kinematic control of redundant manipulators using neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2016.

[26] D. Guo and Y. Zhang, "Li-function activated znn with finite-time convergence applied to redundant-manipulator kinematic control via time-varying jacobian matrix pseudoinversion," *Applied Soft Computing*, vol. 24, pp. 158–168, 2014.

[27] S. Li, H. Cui, Y. Li, B. Liu, and Y. Lou, "Decentralized control of collaborative redundant manipulators with partial command coverage via locally connected recurrent neural networks," *Neural Computing & Applications*, pp. 1–10, 2012.

[28] W. Shen, J. Gu, and Y. Ma, "3d kinematic simulation for pa10-7c robot arm based on vrml," in *Automation and Logistics, 2007 IEEE International Conference on*. IEEE, 2007, pp. 614–619.