Planning of Service Mobile Robot Based on Convolutional LSTM Network

Shuai Yin and Arkady Yuschenko
Department of Robotic Systems and Mechatronics, Bauman Moscow State Technical University, Moscow, 105005, Russian
*Email: yins274@gmail.com and arkadyus@mail.ru

Abstract. This paper proposes a method to complete motion planning in one step using convolutional Long Short-Term Memory (LSTM) network. Service mobile robot movement from the starting position to the target position includes three main tasks: mapping, positioning, and motion planning. The convolutional LSTM network mainly uses the network to complete motion planning. The input of the network is a GRB picture with obstacles, target position, and starting position. The outputs of the network are linear velocity and angular velocity of service mobile robot. The convolution layer of the network is used to mark obstacles, target position, and starting position. LSTM layer describes the time characteristics of movement and full connected layer is used to smoothly fit linear velocity and angular velocity of service mobile robot. The convolutional LSTM network can complete tasks of path finding and control, that is, mapping pictures with obstacles, target position, and starting position to linear velocity and angular velocity of service mobile robot. Compared with traditional separate solutions for motion tasks, this method has obvious advantages such as good fault tolerance and complete motion tasks planning in real time. In the experiment, a mobile robot “Turtlebot” based on ROS systems was used to verify the effectiveness and convenience of the method for motion planning.

1. Introduction
Autonomous mobility systems are widely used in many fields, such as self-driving, autonomous navigation of aircraft and ships, autonomous movement of indoor service robots, etc. Mobile service robots (collaborative robots) are required to work with people. Voice dialogue communication with people is a very effective and convenient way. In the process of dialogue, the robot needs to continuously perform voice recognition and analysis of the dialogue content to determine the task requirements and can ask people questions to supplement the details of the task requirements. There is a detailed analysis of this process in the previous work [1]. Mobile manipulation service robot tasks can be roughly divided into two categories: object grabbing tasks and autonomous navigation tasks. This paper mainly discusses the autonomous navigation task of indoor service robots. The task of grabbing objects has been discussed in the previous work [2]. The autonomous navigation task of the service mobile robot may be divided into two stages: the mapping stage of the autonomous navigation and movement stage according to the prescribed task [3]. The mapping stage result is to build a 2D map or a 3D map of the working environment. The 2D maps were used as a basis for navigation and movement in this paper. In the autonomous navigation and movement stage, the service mobile robot needs to determine the target position based on the content of the dialogue with the user and the result of object detection, and then find a possible path between the target position and its own position, and then generate a series of trajectory points with speed constraints. The photoelectric encoder of motor is
used to calculate the change of the position and posture of the mobile robot. The paper proposes the method that can complete motion planning in one step using convolutional LSTM network. The input of the network is the RGB picture from the position of the mobile robot and the target position, and the output is the linear and angular velocity of the mobile robot. The convolutional LSTM network is used instead of path planning and trajectory generation in traditional methods, reducing a lot of online calculation and optimization time. This method is similar to human control of the mobile robot to move to the target position according to the displayed picture. This method has good fault tolerance and can complete motion planning tasks in real time.

2. Mapping and Positioning

Mapping and positioning are important prerequisites for navigation and control of mobile robot. The task of mapping is to convert the two-dimensional information of sensors into three-dimensional information of the physical world. Mapping and positioning are often carried out at the same time. The coordinate points obtained by the laser sensor are relative to the current position of the sensor. The translation vector and rotation matrix also need to be used to convert the coordinate points to the coordinate points relative to the world coordinate system (see formula 1).

\[
\begin{pmatrix}
  u \\
  v \\
  1
\end{pmatrix}
= \begin{pmatrix}
  f_x & c_x & t_x \\
  0 & f_y & t_y \\
  1 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
  r_{11} & r_{12} & r_{13} & t_x \\
  r_{21} & r_{22} & r_{23} & t_y \\
  r_{31} & r_{32} & r_{33} & t_z \\
  1 & 1 & 1 & 1
\end{pmatrix}
\begin{pmatrix}
  x \\
  y \\
  z
\end{pmatrix}
\]

(1)

where \( u, v \) – pixel coordinates in the picture; \( f_x, f_y, c_x, c_y \) – camera internal parameters; \( r_{11}, ..., r_{33} \) – rotation variables of homogeneous matrix; \( t_x, t_y, t_z \) – translation variables of homogeneous matrix; \( x, y, z \) – the coordinate point relative to the world coordinate system. It can be seen that the determination of the rotation and translation variables of the laser sensor is of great significance to the accuracy of mapping and positioning. A series of such variables are also called odometers. There are many calculation methods for the odometer, which can be obtained by combining the feedback information of the motor, gyro, accelerator and other sensors [4], or by computer vision feature point detection and geometric light principle [5]. The service mobile robot obtains the task requirements in the dialogue with human, such as "Bring the tool at the table near the door to the laboratory table". The convolutional LSTM network is used to control the mobile robot movement to the door position. After reaching the door position, the robot looks for the required tool. After finishing the grasping tool, the robot uses the detection network to find the laboratory table by rotating the RGB-D camera and then move to the laboratory table and put the tools on it.

3. Path Finding and Trajectory Generation

After the map describing obstacles and passable spaces is established (see figure 1a), the mobile robot position is marked as the starting position, and the vicinity of the designated object can be marked as the target position through the object detection algorithm (including object recognition) [6] (see, figure 1b).

The task of path finding is to find a path that can pass under the minimum cost function in the known space. The minimum cost function may be Manhattan distance, Euclidean metric or rules for evaluating obstacles and target locations. Traditional motion planning methods include two parts: path finding and trajectory generation. After path finding, we can get results similar to the figure 1c. It can be seen from the figure 1c that the passable path is made of a limited number of points, and each point has a total of robot position and posture information. But this path is not ideal, and the path needs to be smoothed so that the mobile robot can track the path (see figure 1d).
Trajectory generation is to perform position, velocity and acceleration constraints on a series of points generated by path finding (see figure 1e). Apparently, this means an approximation using splines, at least of the third order. But this may be not enough, since it is necessary to accelerate smoothly and brake smoothly. These are already fourth order splines. Trajectory generation is to provide input for the motor, and the methods are roughly divided as: minimum snap trajectory generation [7], soft and hard constrained trajectory optimization [8]. Trajectory generation task become more complicated for a group of mobile robots [9]

4. Convolutional LSTM Network
The autonomous navigation system using convolutional LSTM network consists of four parts: mapping, positioning, image generation and convolutional LSTM network itself (see figure 2). The mobile robot determines the task requirements according to the content of the dialogue with the human, and then rotates the RGB-D camera to find the target position by object detection algorithm, and transmits the position to the image generator. After completion of this step, the mobile robot determines its position according to the photoelectric encoder information of the motor, the depth image of the RGB-D camera and the map, and sends it to the image generator. The image generator generates an RGB image with the robot position and target position, and then sends the image to the convolutional LSTM network as input. The network generates linear and angular velocity commands to control the robot according to the input picture. The robot moves to the target position according to the linear velocity and angular velocity commands. During the movement, the robot uses the photoelectric encoder and the RGB-D camera to determine its own position change and transmit it to the image generator in real time. The image generator generates a new picture again and sends it to the network. The network generates a new speed command based on the new generated picture, and the robot changes the direction of movement according to the speed command. The above process loops until the robot reaches the target position.

Convolutional LSTM network can map a series of time-characteristic maps marked with the start position and the end position to the speed command of the mobile robot. The network has three main parts: convolution layer, long short-term memory layer and full connected layer. The training of the network needs to be completed in advance. When using it, one only need to input the picture to the network and pass the output linear velocity and angular velocity commands to the motor of robot.

4.1. Convolution Layer
Convolutional layer is an important means of processing pictures, it consists of two sub-layers: conv-layer and pool-layer. Conv-layer uses a certain number of filters to reduce irrelevant information of the picture and enhance the important information of the picture (see formula 2). The pool-layer crops the picture to reduce the processing volume of the LSTM-layer (see formula 3).
where $x_{i,j}$ – feature map element; $f$ – activation function (ReLU); $w_{m,n}$ – filter coefficients; $\alpha_{i+m,j+m}$ – input coefficients; $w_b$ – additional threshold.

The size of the resulting feature map is determined by the formula 3:

$$W_2 = \frac{W_1 + 2P}{S} + 1$$
$$H_2 = \frac{H_1 + 2P}{S} + 1$$

where $W_2$ – width of the resulting feature map; $H_2$ – length of the resulting feature map; $W_1$ – input map width; $H_1$ – input card length; $F$ – filter size; $P$ – the amount of zero-padding around the input image; $S$ – movement step.

Formula of the pool-layer is the next:

$$\delta'_{i,j} = \max(x_{i,j})$$

where $\delta'_{i,j}$ – the feature of map after pooling.

4.2. Flatten-layer

After the image with RGB three channels is processed by the convolutional layer, a $[W \times H \times N]$-dimensional matrix will be obtained. Flatten-layer changes the $[W \times H \times N]$ three-dimensional matrix into a $[1 \times W \times H \times N]$ vector $x$. The $[1 \times W \times H \times N]$ vector $x$ is looked as input of LSTM.

4.3. Long Short-Term Memory Layer

Taking into account that the speed command given to the robot should have time characteristics, that is, the current speed command and the previous speed command have a certain correlation to ensure the continuity and smoothness of the speed command. Long short-term memory layer is used for describe time characteristic.

LSTM-layer have chain like structure in a very special way. It’s very easy for information to just flow along it unchanged (see figure 3).

The key to LSTMs is the cell state $C$, the horizontal line running through the top of the diagram. The cell state $C$ is kind of like a conveyor belt and keep historical information. It runs straight down the entire chain, with only some minor linear interactions.

![Figure 3. Structure of LSTM layer.](image_url)

The output of the LSTM-layer $h_i$ is based on the output $h_{t-1}$ at the $t$-th moment, the cell state $C_i$ at $t$ moment and the current input vector $x_t$ at $t$ moment.

$$f_t = \sigma(w_f [h_{t-1}, x_t] + b_f)$$
$$i_t = \sigma(w_i [h_{t-1}, x_t] + b_i)$$
$$C_t = \tanh(w_c [h_{t-1}, x_t] + b_c)$$
$$h_t = o_t \times \tanh(C_t)$$

$$C_t = f_t \times C_{t-1} + i_t \times C_t$$
$$C_t = f_t \times C_{t-1} + i_t \times \tanh(C_t)$$

where $\sigma$ – activation function sigmoid; $W_f, W_i, W_c, W_o, U_f, U_i, U_c, U_o$ – parameters to be optimized; $b_f, b_i, b_c, b_o$ – additional threshold; $x_t$ – input vector;

The activation function $\sigma$ outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means “let nothing through,” while a value of one means “let everything through!”
4.4. Full Connected Layer

After processing of LSTM layer, the output of LSTM layer has time characteristics, but it still needs to be regressed as speed command of robot by the fully connected layer. Fully connected layer has good regression and classification capabilities. For regression tasks, the last layer of fully connected layer does not need to use the activation function than it is better to use the activation function sigmoid.

The convolutional LSTM network consists of three convolutional layers, one LSTM layer, and three fully connected layers in series.

5. Training of Convolutional LSTM Network

The training of convolutional LSTM network uses offline deep learning, the training of the network needs to be completed in advance. Training data can be created by collecting target positions, robot positions and corresponding linear velocity and angular velocity. When using it, we only need to input the picture to the network and pass the output linear velocity and angular velocity commands to the motors of robot. The quality of training depends on three aspects: loss function, optimization and training data.

5.1. Loss Function

For regression tasks, the loss function is generally set as the mean absolute error or the mean square error (6),(7).

\[
\text{loss} = \frac{1}{m} \sum_{i=1}^{m} |y_{true}^i - y_{pre}^i|
\]

(6)

\[
\text{loss} = \frac{1}{m} \sum_{i=1}^{m} (y_{true}^i - y_{pre}^i)^2
\]

(7)

where \(m\) – is number of samples; \(y_{true}^i, y_{pre}^i\) – are true speed and predicted speed; It is not difficult to find that when the absolute error is greater than 1, the mean absolute square is more effective, and when the absolute error value is less than 1, the square error is more effective. Therefore, combining the advantages of the two can set the loss function as

\[
\text{loss} = \frac{1}{m} \sum_{i=1}^{m} \left\{ \begin{array}{ll}
|y_{true}^i - y_{pre}^i|, & |y_{true}^i - y_{pre}^i| \leq 1 \\
(y_{true}^i - y_{pre}^i)^2, & |y_{true}^i - y_{pre}^i| > 1
\end{array} \right.
\]

(8)

where \(k\) – the amplification factor ratio;

5.2. Optimization

The general optimization method for deep learning is the gradient descent method. This method has a faster convergence rate but is prone to oscillations. The adaptive learning rate method RMSprop has obvious effects on regression tasks. The necessary calculations are the next

Gradient calculation:

\[
g \leftarrow \frac{1}{m} \nabla \sum_{i} \text{loss}(f(x^{(i)}; w), y_{true}^i)
\]

(9)

Cumulative squared gradient:

\[r \leftarrow pr + (1 - p)g \odot g \]

(10)

Calculation parameter update

\[
\Delta w = -\frac{\epsilon}{\sqrt{\delta + r}} \odot g
\]

(11)

Parameter update

\[w \leftarrow w + \Delta w \]

(12)

where \(r\) – cumulative variable; \(p\) – decay rate; \(g\) – gradient value; \(m\) – number of samples; \(\epsilon\) – global learning rate; \(\delta\) – small constant; \(\odot\) – multiply elements; \(x^{(i)}\) – input picture; \(w\) – network coefficient;

5.3. Training Data

After completing the network architecture and setting the loss function optimization method, it is necessary to collect training data and train the network. The training and the using of the network are
completely independent. Only when the training is completed and a better network is obtained, we can it perform well in use.

Suppose the input of the convolutional LSTM network is a matrix of size 224*224*3, and the output is linear velocity and angular velocity of 1*2 size. Let generate the training data by recording the speed commands of the mobile robot and the space coordinates of the mobile robot using traditional methods (path finding and trajectory generation). The sliding window method is used to process training data. Let us take 5 sampling times as the sliding window length, and shift 4 sampling times each time. In the paper, the grid is set to 10*10. Each set of starting point and ending point can generate nearly 200 input images and 200 corresponding output speeds.

The sampling time should be proportional to the positioning interval. Too large sampling time will make the network lose its time characteristics and be hard to converge. Too small sampling time will increase the difficulty of training and decrease the accuracy. After many experiments, it was found that when the sampling time is set as slightly larger than the positioning time interval, the network regression accuracy is higher and the convergence speed is faster. An example is illustrated on figure 4a. The accuracy of the convolutional LSTM network regression is shown by the mean absolute error evolution (see figure 4b). At the beginning of the iteration, the network converges quickly. When iterating 25 times, the mean absolute error is 0.02 and not significantly reduced. When iterating 175 times, overfitting occurs.

Dynamic model and kinematics model of two-wheel differential speed are not described in detail here. In the GAZEBO application of the ROS system, only the physical parameters (mass, volume, shape, etc.) of the robot's wheels and car body need to be set.

6. Experiment

The RGB-D camera Kinect was used to build 2D map and to determine robot location in the experiment. GAZEBO was used to simulate the dynamic model of mobile robots and obstacles in the physical world. The trained convolutional LSTM network model was used to replace the global and local motion planning of the mobile robot Turtlebot. The mobile robot is positioned by the RGB-D camera Kinect, and the coordinates and rotation angle of the robot in the 2D map are transmitted to the image generator. The image generator marks the robot coordinates as a green circle and marks the direction of the robot as a green straight line. The target coordinates and direction are marked in red. Because the computer simulation physical environment occupies a part of the CPU, a new ROS node is created to calculate the convolutional LSTM network in the GPU 1080ti to improve the calculation speed of the entire simulation system. Figures 5 and 6 show the experimental results.
Figure 5. Simulation results of using convolutional LSTM network to control mobile robot in ROS system.

In the figure 5, a) the green is the starting position and direction of the robot, the blue is the movement track, and the red is the target position and direction. b) it is the C space of the 2d map that RIZE in ROS displays in real time. c) shows the real robot and working environment simulated by GAZEBO. d) is the linear velocity (red point) and angular velocity (grey point) commands output by the convolutional LSTM network to the mobile robot. More experimental results can be seen in the figure 6.

The fault tolerance of the network is directly related to the diversity of training data. If the training data is single, the network has low fault tolerance and does not have the ability of generalized motion planning. From the other side the large amount of disordered training data makes the network unsuitable for convergence. Therefore, the 2D map can be divided into grids, and training data can be collected at each grid point that can be set to start and target locations. The training data obtained in this way can make the network have good fault tolerance (see Figure 6 g, h).

7. Conclusion
Let repeat the main stage of the proposed procedure. First of all, the goal is set and adjusted by means of a speech dialogue with the operator. Then the map is built and the trajectory is planned on the map. To complete the path planning, the motion trajectory is generated and the speed command is calculated to control the mobile robot to move to the target position. The method with using convolutional LSTM network can realize the calculation of speed command in one-step to control the robot to move to the target position. Experiments verify that the method can generate smooth speed commands in real time to control robot to move to the target position accurately and has good fault tolerance. We suppose that the method proposed above may be applied also for more complicated tasks such as the planning the routes for a group of mobile robots [9] and in the case of moving obstacles [10].

Reference
[1] Yin S, Yuschenko A 2019 The application of the convolutional neural network to organize the work of a collaborative robot - surgeon assistant Interactive Collaborative Robotics, 11659 287-297
[2] Yin S, Yuschenko A 2019 Object Recognition of the Robotic System with Using a Parallel Convolutional Neural Network Industry 40 Issues & New Intelligent Control Paradigms, 272, 3-11
[3] Mikhailov B B, Nazarova AV, Yuschenko AS 2016 Autonomous Mobile Robots Navigation and Control Izvestiya SFedU, Engineering Sciences, 2(175) 48-67
[4] Wang X, Lian Y and Li L 2018 Localization of Autonomous Cars Using Multi-Sensor Data Fusion
*Chinese Automation Congress*, 2018 4152-4155

[5] Mikhailov BB, Devetjarikov EA 2012 Visual odometer *Vestnik BMSTU, Priborostroenie*, special issue № 6 68-82

[6] Zhao ZQ, Zheng P, Xu S and Wu X 2019 Object Detection with Deep Learning: A Review *IEEE Transactions on Neural Networks and Learning Systems* 30, 3212-3232

[7] Mellinger D and Kumar V 2011 Minimum snap trajectory generation and control for quadrotors *IEEE International Conference on Robotics and Automation*

[8] Carius J, Ranftl R, Koltun V and Hutter M 2018 Trajectory Optimization with Implicit Hard Contacts *IEEE Robotics and Automation Letters*, 3 3326-3323

[9] Zenkevich S L, Nazarova A V, Zhu H 2019 Logical Control a Group of Mobile Robots
*Gorodetskiy A, Tarasova I (eds) Smart Electromechanical Systems Studies in Systems, Decision and Control, Springer* 174 32-43

[10] Mikhailov B B, Gerasimov V N 2012 Mobile robot movement control in the case of moving obstacles *Vestnik BMSTU, Priborostroenie*, special issue № 6, 83-92