PATH PLANNING USING VREP

K. Karteek Reddy[1], K.Praveen[2]

[1, 2]*Department of Mechanical Engineering, M.V.G.R. College of Engineering, JNTU (Kakinada Vizianagaram, Andhra Pradesh, India,* ***karteek968@gmail.com, praveenkalla@mvgrce.edu.in***

Abstract

This paper presents an overview of path planning of a mobile robot using VREP [Virtual Robot Experimentation Platform] which is a powerful m3D simulator that create, compose, simulate any robot and is said to be a Swiss army knife among robot simulators. This allows for versatile prototyping applications including systems verification, safety/remote monitoring, rapid algorithm development, the robot model can be added by various entities and primitive shapes and sensors which include proximity sensor, vision sensor, and force sensor with different shapes.

***Index terms*:** *Path planning, VREP, Mobile robot*

----------***----------

1. INTRODUCTION

The overall architecture and control methodology are crucial elements in robot simulators. Now a days the robots are employing with highly complex control methods when compared to the robots in the previous generations ,as well as the processing power of computers also increases, A good simulation requires careful architecture to get both performance and accurate calculations (particularly with respect to dynamic conditions). For example, the dynamics model is used rarely when the kinematics is failed on V-REP's control methodology; in V-REP, control is distributed and based on an unlimited number of scripts that are directly associated or attached to scene objects. Finally, in order to further clarify elements introduced in the first and second part of this paper, the third party describes via typical VREP simulation software.

2. CALCULATION MODULES

2.1 Inverse Kinematics

Allowing handling any type of mechanisms i e branched, closed, redundant, containing nested loops. The module is based on calculation of the damped least squares pseudo inverse [1]. It supports conditional solving, damped and weighted resolution, and obstacle avoidance based constraints.

2.2 Collision Detections

Allowing fast interference checking between any geometry or collection of geometries with optimal collision contour calculation The module uses data structures based on a binary tree of Oriented Bounding Boxes [3] for accelerations. Additional optimization is achieved with a temporal coherency caching technique

2.3 Path Planning

Allowing holonomic and non holonomic path planning tasks

2.4 Minimum Distance Calculations

Allows fast minimum distance calculations between any geometries ie convex, concave, open or closed or collection of geometries. The module uses the same data structures as the collision detection module. Additional optimization is also achieved with a temporal coherency caching technique

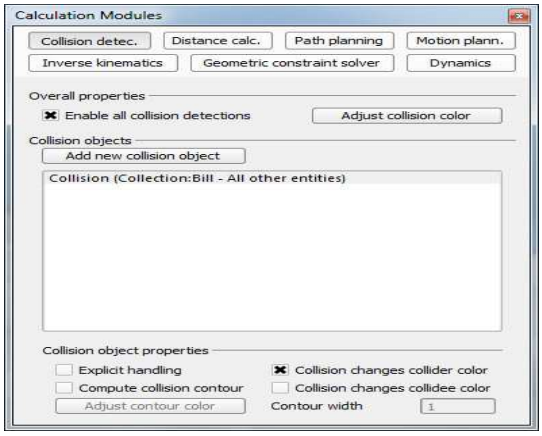

Fig1 Calculation module dialog box

2.5 Physics/Dynamics

Allows handling rigid body dynamics calculation and interaction (collision response, grasping, etc.) via the Bullet Physics Library [7]

2.6 Geometric Constraint Solver Module

Allowing to intuitively to solve/actuate/interact with mechanism .It is a perfect tool for fast prototyping and verification .The models are available in this simulator itself or we can import the models by designing in the modeling software's which may be holonomic or non holonomic robots and is also for remote

monitoring, fast algorithm development, robotics related education and simulation of factory automation

3. SCENE OBJECTS

A V-REP simulation scene contains several scene objects or elemental objects. The following scene objects are supported in V-REP:

3.1 Shapes

Shapes are triangular meshes, used for rigid body visualization.

3.2 Cameras and Lights

Cameras and lights are used for scene visualization purposes mainly, but can also have effects on other scene objects (e.g. lights directly influence rendering sensors). Sensors make use of hard- ware acceleration for the raw image acquisition

3.3 Dummies

Dummies are \points with orientation," or reference frames, that can be used for various tasks, and are mainly used in conjunction with other scene objects,

3.4 Proximity Sensors

The proximity sensors objects perform an exact minimum distance calculation within a given detection [4]) as opposed to simply performing collision detection between some selected sensing rays and the environment; hence allowing for reactance effects due to sensor/surface angles.

3.5 Rendering sensors:

Rendering sensors In V-REP are camera-like sensors, allowing to extract complex image information from a simulation scene (colors, object sizes, depth maps, etc lter elements via plug ins). Rendering

3.6 Force Sensors

Force sensors are rigid links between shapes, that can record applied forces and torques, and that can conditionally saturate

3.7 Graphs

Graphs are scene objects that can record a large variety of one dimensional data streams. Data streams can be displayed directly (time graph of a given data type), or combined with each other to display X/Y graphs,
or 3D curves.

3.8 Mills

Mills are customizable convex volumes that can be used to simulate surface cutting operations on shapes (e.g., milling, laser cutting

3.9 Paths

Paths allow complex movement definitions in space (succession of freely combinable translations, rotations and/or pauses), and are used for guiding a welding robot's torch along a predefined trajectory, or for allowing conveyor belt movements

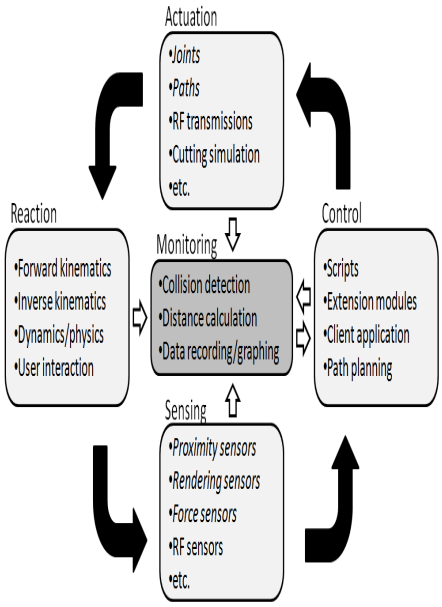

Fig2 Simulation loop in VREP

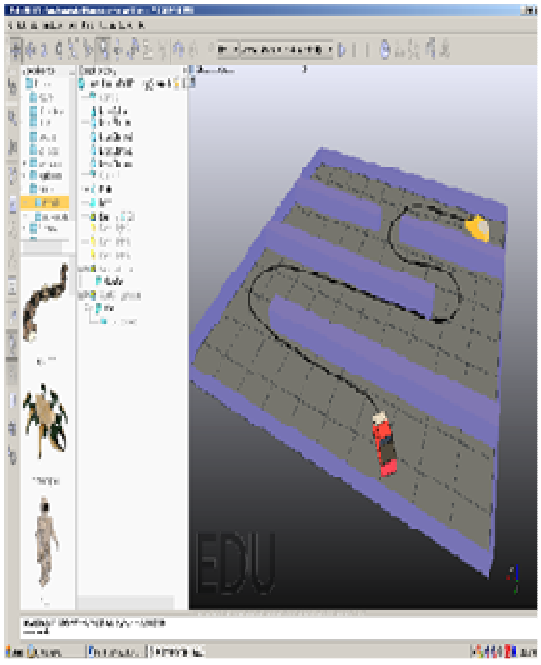

Fig 3 Path planning model created by elements

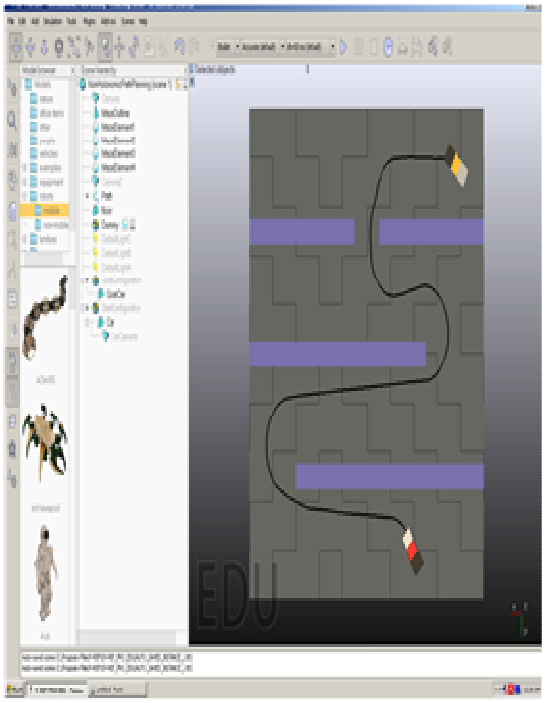

Fig4: Another view of the scene

4. SIMULATION PROCESS

A simulation in V-REP can be started, paused and stopped. The simulation can be done by using the bullet dynamics engine, ODE dynamics engine and real time simulation [7]. Each time the main script was executed, the simulation time is incremented by the simulation time step. Using large time steps results in fast but inaccurate/unstable simulations. During real-time simulation, it can happen that the simulation time is not able to follow the real-time (e.g. because of some momentarily heavy calculations). In that case, if this check-box is selected, then the simulation time will try catching up the lost time (e.g. when the calculation load is again reduced), which results in an apparent speed-up A path linking the start configuration to the goal configuration can be specified (or restricted to be) in a configuration space with a specific number of dimensions (e.g. the X, Y configuration space).[6]. Moreover additional constraints are usually needed that make the task more complicated (e.g. keeping a certain distance threshold to the obstacles, or moving only in one direction A page in V-REP is the main viewing surface of a scene. It is not directly a view, but can contain one, two, or as so many views as needed.

A view is what is used to display the image content of a specific object which has to be a viewable object. If a view is associated with a camera object for instance, then it can display what the camera sees. Following figure explains the page, view and viewable object relationship.

The simulator operates by advancing the simulation time at constant time steps. Following figure illustrates the main simulation loop: positions of the model with respect to axes

The table shows the orientations and position of different elements used to construct a model

Blocks	X	Y	Z
MAZE	0.1	0.25	0.025
MAZE	0.275	-0.25	0.025
MAZE	-0.275	-0.25.	0.025
MAZE	0.1250	0	0.025
START CONFIGURATION	-0.1919	0.4010	0.025
GOAL CONFIGURATION	-0.3795	-0.3795	0.0251

5. SCRIPTS

In V-REP, each script has a list of simulation parameters. Those parameters can be used as a quick way of adjusting values of a specific model for example (e.g. the maximum velocity of a mobile robot)[2]. The script simulation parameters for a script can be accessed by double-clicking the appropriate icon in the scene hierarchy: script simulation parameters are also used for specific inter-script messaging purpose

5.1 Initialization Section

The initialization section should be run only the first time the script or program is run. , however: remember, scene objects as well as child scripts

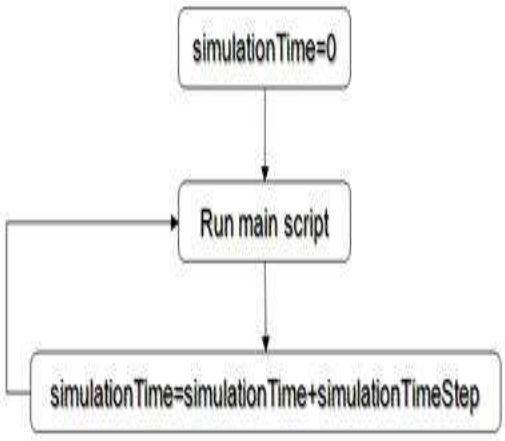

Fig5 Main simulation loop

It can be copy/pasted into a scene at any time, even when a simulation is running.

5.2 Simulation Section

The simulation section should always be run and n the simulation section should always be run and no specific state monitoring is required except for threaded child scripts. Threaded child scripts should make sure to end execution If not, the simulator will wait longer than needed before being able to end a simulation, and the restoration code of the threaded child script might not be run

5.3 Restoration Section:

The restoration section should be run just before a simulation ends. Threaded child scripts on the other hand do not need to monitor any state for the restoration: the restoration section is simply the last part of the code

6. Non threaded child scripts

Non-threaded child scripts are pass-through scripts. This means that every time they are called, they should perform some task and then return control to the calling script[5]. If control is not returned to the calling script, then the whole simulation halts. They operate as functions and normally non threaded child scripts are called at each simulation step (each time with a different simulation time). This type of child script should always be used over threaded child scripts whenever possible.

7. Threaded child scripts

Threaded child scripts are scripts that will launch in a thread. This means that a call to a threaded child script will launch a new thread and then directly return[8]. The new thread will then execute in parallel. When a threaded child script was already launched (and didn't end yet) but is called again, then the call is ignored. An execute once option is set by default. There can be as many threaded child scripts running as needed. A threaded child script icon in the scene hierarchy

Threaded child scripts can in their turn call non-threaded as well as threaded child scripts. When a non-threaded child script is called from a thread that is not the main thread, then the child script is said to be running in a thread.

Threaded child scripts have several weaknesses compared to non-threaded child scripts if not programmed appropriately: they are more resource-intensive, they can waste some processing time, and they can be a little bit less responsive to a simulation stop command.

CONCLUSIONS

V-REP demonstrates a path planning task by framing the model and the simulation is run by setting the time interval, choosing the simulation ie real time , ODE, bullet. This results in a versatile and scalable framework that _ts the simulation needs of complex robotic systems is its distributed control approach; it is possible to test configurations with 2, 3, or up to several hundreds of (identical or different) robots. Thus the path planning of a robot using this simulator is very fast and we can observe keenly by cameras provided

REFERENCES

[1] Wampler, C.W.: Manipulator inverse kinematic solutions based on vector formulations and damped least squares methods. IEEE Trans. Syst., Man, Cybern. 16(1), 93{101 (1986)
[2] Lua, http://www.lua.org
[3] Gottschalk, S., Lin, M.C., Manocha, D.: OBB-tree : a hierarchical structure for rapid interference detection. In: ACM SIGGRAPH. pp. 171{180. New Orleans, USA
(October 1996)
[4]Freese, M., Ozaki, F., Matsuhira, N.: Collision detection, distance calculation and proximity sensor simulation using oriented bounding box trees. In: 4th International Conference on Advanced Mechatronics. pp. 13{18. Asahikawa, Japan (October 2004)
[5] Marc Freese*, Surya Singh, Fumio Ozaki, and Nobuto Matsuhira *K-Team Corporation, Y-Parc - Rue Galil_ee 9, 1400 Yverdon-les-Bains, Switzerland mfreese@gmx.ch,spns@acfr.usyd.edu.au,
[6] Ku_ner Jr., J.J.: RRT-connect: an e_cient approach to single-query path planning. In: IEEE International Conference on Robotics and Automation. pp. 995{1001. San Fransisco, USA (April 2000)
[7] Bullet physics library, http://www.bulletphysics.org
[8] Ando, N., Suehiro, T., Kitagaki, K., Kotoku, T., Yoon, W.: RT-component object model in RT-middleware - distributed component middleware for RT (robot technology). In: 2005 IEEE International Symposium on Computational Intelligence in= Robotics and Automation (CIRA2005). pp. 457{462. Espoo, Finland (June 2005)