A Bio-inspired Intelligent Approach to Motion Planning for Mobile Robots

S. H. Choi[1] and W. K. Zhu[2]

The University of Hong Kong
[1]shchoi@hku.hk , [2]wkzhu@hku.hk

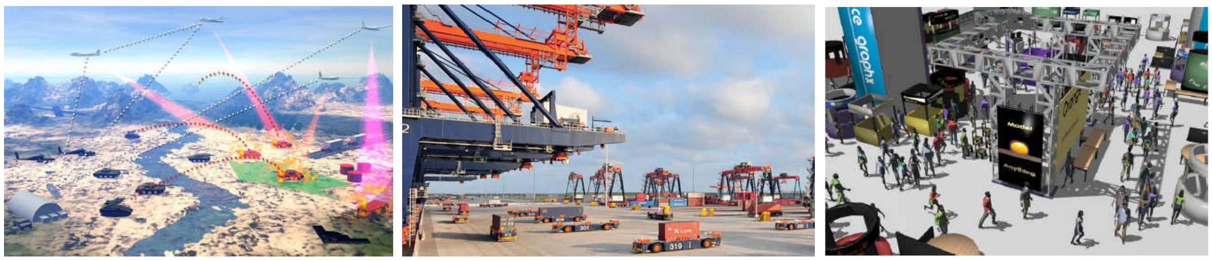
ABSTRACT

Although huge potentials abound for mobile robots in areas like military, traffic control, logistics, and computer graphics, practicable motion planning techniques have yet to be developed. This paper proposes a bio-inspired intelligent approach to motion planning for decentralised mobile objects in dynamic environments. It is inspired by the natural behaviours of creatures which tend to keep a safe distance between one another, and move towards their respective destinations. The proposed approach does not require any central controller, and there is no communication between robots. It instead imitates the characteristics of creatures with local sensing for detection of imminent neighbours and navigation. Each robot is assumed to be driven by a virtual attractive force of its destination and repulsive forces of its imminent neighbours. In particular, it features a module to detect imminent neighbours, reducing computation overheads and eliminating redundant robot movements. Moreover, in comparison with other methods that are mostly based on a simple function for virtual force calculation, the proposed approach adopts a more adaptive two-section function to calculate repulsive forces to improve collision avoidance. A simulator, with a case study of motion planning for free-range automated guided vehicles at a container terminal, is developed to implement and validate the proposed approach.

Keywords: bio-inspired, mobile robots, motion planning, dynamic environments.
DOI: 10.3722/cadaps.2011.773-783

1 INTRODUCTION

Multi-robot motion planning addresses the problem of how a team of autonomous mobile robots can share the same workspace, while avoiding interference with each other and achieving group motion objectives [10]. Research on this field dates back to the late 1980s, initially beginning with the study of mobile robotics. Motion planning for multiple autonomous robots to navigate safely and avoid moving obstacles in dynamic environments is still among the most difficult and important problems in multi-robot control, and it is necessary for a number of real world applications, such as military, traffic control, logistics, and computer graphics [6], as shown in Fig. 1.



(a) (b) (c)

Fig. 1: (a) Motion planning for distributed heterogeneous military systems [6], (b) Automated guide vehicles at a container terminal [16], (c) Real-time simulation of realistic motion of multi-agents in computer graphics [17].

The European Union has sponsored several swarm robot projects. The I-SWARM project, for instance, aimed to develop a swarm of robots to perform cooperative tasks, such as foraging. An inter-university SWARMS initiative in the United States tried to develop a new system framework for controlling a swarm of mobile robots, synthesizing emergent behaviours for reactive response, and developing algorithms for decentralised transport [10].

The environments of motion planning can be divided into two categories, namely static and dynamic. Motion planning approaches for static environments plan routes of robots from origins to destinations from completely known information of the environments and the routes are repeatedly used if the same tasks are assigned. Obviously, static approaches are not able to adapt to changes in the system and traffic conditions. Dynamic motion planning approaches generate routing decisions based on constantly changing real-time information and the routes of robots are continually adjusted, segment by segment. The advantage is that validity of routes can be achieved during operation, even under dynamic traffic conditions and task requests. However, the optimality of the complete routes cannot be guaranteed [18].

The architecture of motion planning for a team of robots can be centralised or decentralised [11]. A centralised architecture uses a single decision maker to plan the motions of all robots. In theory, it is easy to implement, and it can produce optimal solutions by gathering all relevant information. In practice, however, it suffers obvious limitations. A large amount of resources are required for the central controller to handle all the computation tasks as well as frequent communications with all robots, which often incur sluggish response of individual robots to constantly changing local environments. Furthermore, the whole team will become paralytic if the central controller fails. Thus, centralised control tends to suit applications involving small number of robots in static environments with easily available global information. A decentralised architecture, on the other hand, lets each robot make its own decisions based only on its local information, without any communication with and intervention from a central controller. Therefore, immediate and flexible response to dynamic local changes can be achieved. This type of architecture is more fault-tolerant, for it is affected by possible failures of a central controller and even if some robots fail, the rest of the team can still function normally. Moreover, a decentralised system is well scalable for a large number of robots, because the decision-making of each robot concerns only with its own local environment which is independent of the total number of robots in a team [7][19]. Nevertheless, optimal decentralised solution of each robot may not necessarily aggregate to an optimal global solution of the team. Hence, the challenge of decentralised motion planning is to predict the collective performance based on individual decision making, and vice versa [9]. This paper adopts a decentralised architecture to plan the motions of a robot team in dynamic environments.

A number of open issues in multi-robot motion planning remain. Dynamic re-planning of motion is important in stochastic environments, and most of the current techniques still have difficulty in handling environment uncertainties, such as stochastic task requests and unexpected traffic conditions. Secondly, existing techniques typically cannot scale well to handle a large number of robots, and there are limitations for extensions to three-dimension applications like aerial robots. Thirdly, developing motion planning techniques that incorporate practical motion, sensing, and communication constraints of physical robots is also desirable [10].

There are some popular techniques of multi-robot motion planning, such as virtual forces approaches [4], potential field methods [19], mixed integer linear programming [14], protocol-based [8], genetic algorithms [13], spatiotemporal planning [2], and graph-based [11], etc. Each of them has its merits and deficiencies, and is applicable only in a limited range of applications.

Of these techniques, the virtual forces approaches are among the effective methods to handle dynamic situations. The principle of these approaches assumes that each agent is influenced by virtual forces from other individuals or its destination, according to the applications concerned and design criteria. Helbing et al. [4] proposed an empirically derived social forces model to simulate the dynamics and motion of pedestrian crowds. This model described the psychological tendency of pedestrians to stay away from each other by a repulsive interaction force. In case of contacts between pedestrians, a body force counteracting body compression and a sliding friction force impeding relative tangential motion were also proposed, which were inspired by granular interactions. This model was estimated from real data. Scenarios of human crowds, particularly evacuation of panic crowds, were presented. The disadvantage of this social forces model was that agents appeared to shake in response to numerous forces in high-density crowds, and the agents were only modelled as particles. Silveira et al. [15] used a virtual forces model to control behaviours of mobile agents. The agents can negotiate the respective motion, avoid collisions, and attain respective goals, while producing very individual paths. The individuality of each agent could be set by changing its inner force parameters, leading to a broad range of possible behaviours without jeopardizing its performance. Examples of steering behaviours in corridors with collision avoidance, and searching for objects in unknown environments were presented. The agents with this approach could only move in a grid-based map without consideration of motion dynamics, while the obstacles remained static. These virtual forces approaches are easy to understand, with clear mathematic formulation, and most importantly, reactive to uncertainties of environment. And yet, they suffer some limitations. The planning parameters, such as sensing range and rate, force calculation, and system dynamics, are usually empirical, determined by trial and error. Also, there is a lack of theoretical analysis of system stability and convergence [1].

Similar approaches are potential field methods, which are physics-inspired. The potential field methods direct robots as if they were particles moving in a potential vector field. Gradients can be regarded as forces exerted on positively charged robots which are attracted to the negatively charged goals. Obstacles, on the other hand, are positively charged to repel the robots away. Zavlanos et al. [19] developed a distributed multi-destination potential field which was able to drive a swarm of agents to available destinations by the respective attractive forces. A neighbour coordination protocol, facilitated with communications and sensing of agents, was also developed to ensure that local interferences of agents could be possibly avoided. However, it seemed that avoidance of collisions between agents was not sufficiently guaranteed. These potential field methods require full knowledge of the environments prior to motion planning, which is often not known in dynamic situations. Local minimum is another common drawback of these methods. Traditional potential field methods only consider the relative positions of robots. A complementary approach, called the generalised potential field method, also considers the velocities of robots. However, the complementary forces only affect in the directions of velocities and cannot steer the yaw angles [1].

To meet these challenges of multi-robot motion planning, this paper aims to develop a bio-inspired intelligent approach to motion planning for a team of decentralised mobile robots working in a dynamic environment. Intelligent decentralised motion planning in a dynamic environment can be observed from natural creatures, such as a flock of animals, and human pedestrians in a crowd. These creatures maintain local sensing to spot imminent neighbours without explicit communications; they tend to keep an adequate distance from one another, effectively solving local interferences and moving towards their respective destinations.

This paper assumes that every mobile robot in a team keeps sensing it local environment to detect imminent neighbours which are likely to pose immediate collision dangers. It is virtually separated from its imminent neighbours by repulsive forces, while approaching towards its destination by an attractive force. To describe the dynamics of these virtual forces on the motion of each robot, the subsequent navigation function is derived in the form of Newton's law of motion. Unlike most of the similar navigation functions that simply take all the sensed neighbours into account, the proposed one features a module to detect imminent neighbours. With this detection module, only those robots likely to collide will be processed accordingly at the following navigation stage, thus reducing computation overheads and eliminating redundant robot movements. Moreover, while most other methods are based on a simple function for virtual force calculation, we adopt a more adaptive two-section function to calculate repulsive forces to improve collision avoidance. The proposed approach is implemented in a simulator developed for a case study of motion planning for free-range automated guided vehicles at a container terminal.

2 THE BIO-INSPIRED INTELLIGENT MOTION PLANNING APPROACH

As mentioned above, the proposed motion planning approach to multiple mobile robots is inspired by the motion behaviour of natural creatures. A motion planning cycle of a robot equipped with modern advanced processor and sensors includes three stages, namely local sensing, detecting imminent neighbours, and real-time navigation. Fig. 2 shows the flow of such a cycle, which can be sufficiently short in order to avoid obsolete information. The motion planning approach works in an incremental pattern such that the route is planned segment by segment until the robot reaches its destination. The advantage is that the validity of routing can be achieved during operation even in dynamic traffic conditions.

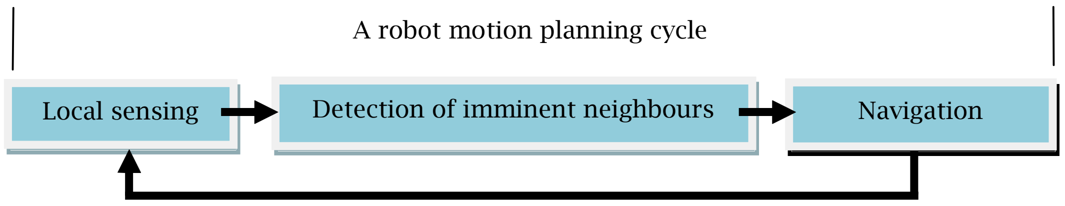

A robot motion planning cycle
Local sensing
Detection of imminent neighbours
Navigation

Fig. 2: Three stages in a robot motion planning cycle.

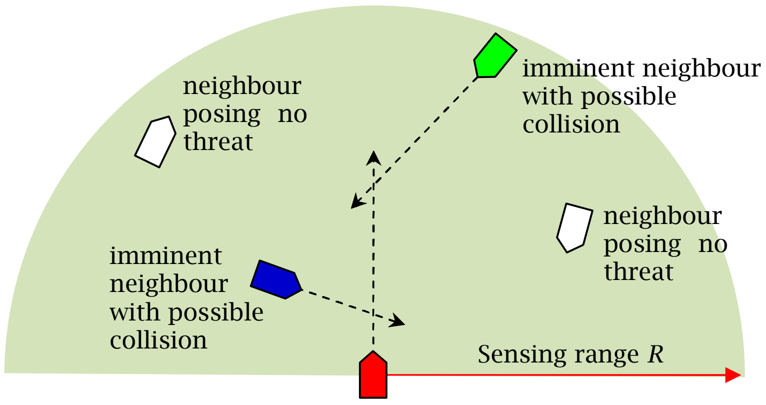

neighbour posing no threat
imminent neighbour with possible collision
neighbour posing no threat
imminent neighbour with possible collision
Sensing range R

Fig. 3: Local sensing and detection of imminent neighbours by a robot with a 180° field of view.

2.1 Local Sensing

The field of view of a range sensor, such as a sonar or a laser sensor, can be 180°, 360°, or other values. In this paper, a 180° field of view is used, as shown by the red robot in Fig. 3. This is similar to a lot of creatures. They need not observe the situations behind, since the neighbours behind are also sensing and avoiding collision with their front neighbours. Moreover, a scanning view of 180° is usually twice as efficient as 360° scanning. Another issue of local sensing is the scanning range *R*, which is subject to the requirements of the application concerned, and the technical specification of a sensor.

2.2 Detection of Imminent Neighbours

Most motion planning methods discussed in Section 1 simply take all the sensed neighbours into account. Such an approach is not only computationally expensive but also results in undesirable robot movements. The proposed motion planning approach, on the other hand, features a detection module to spot imminent neighbours which pose immediate dangers to the robot concerned. Incorporating this module, the resulting navigation of a robot should outperform the existing ones.

Fig. 3 shows a robot motion planning cycle in which the red robot senses four neighbours. Detection of imminent neighbours is achieved by computing possible intersections of robot motions in range. Since the cycle time is short, it is reasonable to assume that a sensed neighbour robot continues to move in a straight line, as shown in Fig. 4, where $(\vec{s}_{j-1}, t_{j-1})$ and $(\vec{s}_j, t_j)$ represent two consecutive states of a sensed neighbour robot, while $\vec{s}$ stands for position and t for time. Variables with symbol $\rightarrow$ are vectors.

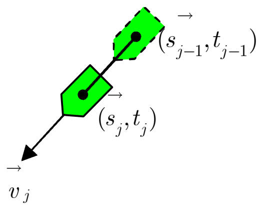


Fig. 4: Straight motion assumed in a robot motion planning cycle

The robot velocity is derived by:

$$\vec{v}_j = \frac{\vec{s}_j - \vec{s}_{j-1}}{t_j - t_{j-1}} \tag{2.1}$$

Referring to Fig. 3, if the extended velocity vector of a robot *i* intersects with that of the red robot, then robot *i*, within the sensing range, is an imminent neighbour to the red robot. It can be expressed as:

$$g(i) = \begin{cases} 1\ , & i \text{ is imminent} \\ 0\ , & \text{otherwise} \end{cases} \tag{2.2}$$

where $g(i)$ is the detection function to be incorporated in the following navigation function. Hence, the blue and the green robots are considered as imminent neighbours that are likely to collide with the red robot. With this detection module, only such imminent neighbours posing immediate threats will be processed accordingly at the navigation stage. This reduces computation overheads and eliminates redundant robot movements. It should be noted that the number and locations of the spotted imminent neighbours are different in most motion planning cycles, because robots are all moving about.

2.3 The Navigation Function

Based on sensing the local environment and detection of imminent neighbours, a navigation function is developed to guide a robot to its destination and to solve local interference between imminent neighbours. It is assumed that each robot is driven by a virtual attractive force of its destination and by some virtual repulsive forces of its imminent neighbours to keep an adequate distance from one another. To describe the dynamics of these virtual forces on the motion of each robot, the navigation function is derived in the form of Newton's law of motion:

$$\vec{f}_{net} = \frac{d(m\,\vec{v})}{dt} = \vec{f}_{goal} - \sum \vec{f}_i \tag{2.3}$$

where $\vec{f}_{net}$ is the net force, $\vec{f}_{goal}$ is the attractive force by the destination, while $\sum \vec{f}_i$ is the composed repulsive force by its imminent neighbours.

To calculate the attractive force by destination, it is necessary to alleviate overshoot and subsequently fluctuations when a robot approaches very close to its destination. Its magnitude is therefore assumed to be proportional to the distance from the goal, as follows:

$$\vec{f}_{goal} = k\,\vec{d}_{goal} \tag{2.4}$$

where $\vec{d}_{goal}$ is the distance vector pointing from the robot to its goal, and k is a scaling factor.

To enhance the responsiveness of a robot to different threats of collision posed by its imminent neighbours, a two-section function is formulated to calculate the repulsive forces. When an imminent neighbour is detected at the outer half of the sensing range, an inversely proportional function is used to mildly steer the robot:

$$\vec{f}_i = \frac{A}{\vec{d}_i}, \qquad \text{if } \frac{R}{2} < d_i \le R \tag{2.5}$$

where $\vec{d}_i$ is the distance vector pointing from an imminent neighbour to the robot, and A is a scaling factor. When an imminent neighbour is detected at the inner half of the sensing range, an exponential function is introduced to exert tight repulsion on the robot:

$$\vec{f}_i = Be^{\frac{-\vec{d}_i}{C}}, \quad \text{if } 0 < d_i \le \frac{R}{2} \tag{2.6}$$

where B is a constant giving the largest repulsive force, and C indicates the changing rate. Like most of the similar approaches mentioned in Section 1, the scaling factors and parameters in force calculation are usually determined by trial and error. Incorporating the detection function (2.2), the complete function of a repulsive force is:

$$\vec{f}_i = \begin{cases} g(i)\bullet\dfrac{A}{\vec{d}_i}, & \text{if } \dfrac{R}{2} < d_i \le R \\[2ex] g(i)\bullet Be^{\frac{-\vec{d}_i}{C}}, & \text{if } 0 < d_i \le \dfrac{R}{2} \end{cases} \tag{2.7}$$

Take the situation in Fig. 3 for example. The red robot is attracted by its destination while being repelled by its imminent neighbours. The net force calculation is shown in Fig. 5. In this motion planning cycle, the red robot steers to the right by the net force $\vec{f}_{net}$.

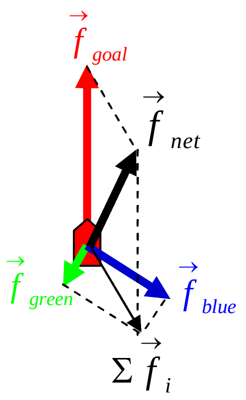


Fig. 5: Net force calculation of the case in Fig. 3

3 IMPLEMENTATION AND CASE STUDY

The proposed motion planning approach is incorporated and validated in a simulator for simulation of a team of autonomous robots transporting containers at a container terminal. The simulator is developed on the Player/Stage [12] open-source platform, which is widely used for multi-robot control and simulation. It consists of two sub-packages, namely Player, and Stage. Player provides a network interface to a variety of physical robots and sensors. Player's client/server model allows robot control programs to be written in a number of programming languages and to run on any computer with a network connection to physical robots. Control program communicates with Player over TCP sockets, reads data from sensors, and writes commands to actuators. Stage is a Player plug-in simulation package which simulates a population of mobile robots moving and sensing in a 2D bitmapped environment. Various sensor models are provided, including sonars, laser rangefinders, pan-tilt-zoom cameras and odometers. Virtual devices of Stage present a standard Player interface, and hence few or no changes are required to move between simulation and hardware. Controllers designed in Stage have been demonstrated to work on various physical robots. In this paper, the Player/Stage runs in a Linux-based operating system called Fedora 13, and the programming language is C++.

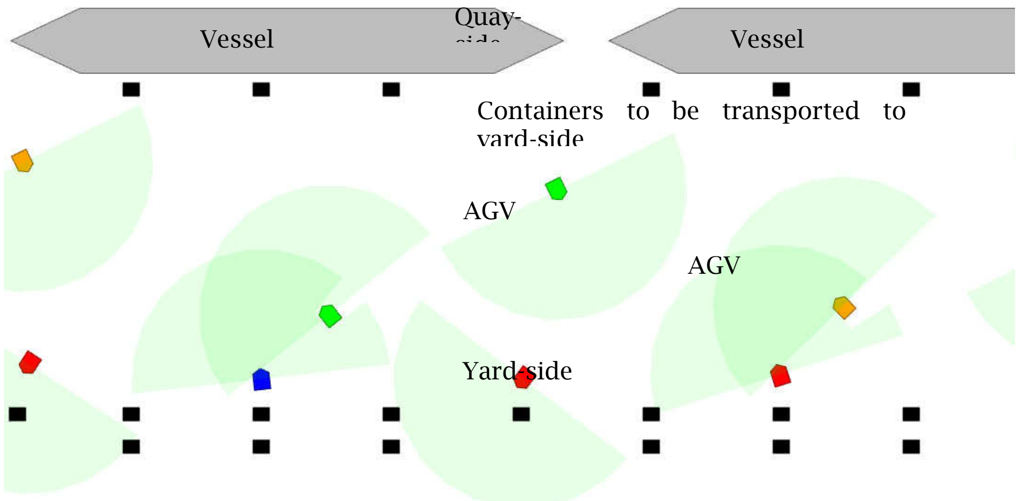


Fig. 6: Simulated working environment of free-range AGVs at a container terminal

An automated guided vehicle (AGV), with autonomous control and sensing devices, can be regarded as an autonomous robot. A team of AGVs at a container terminal transporting containers from the quay-side to the yard-side is used to verify the practicability of the proposed motion planning approach, as shown in Fig. 6. There are two vessels berthed at the quay-side. Each vessel is served by five quay cranes which unload the containers from the vessels. Containers appearing beside the vessel are ready to be picked up, while those being handled by the quay cranes are not shown in the figure. The team of AGVs are transporting containers from the quay-side to the yard-side.

Traditionally, either in manufacturing areas or container terminals, AGVs use fixed guide-paths, such as loops, and networks. The fixed routing approaches allow for reliable automation of vehicles. However, AGVs are less manoeuvrable. Routes are unnecessarily long, incurring considerable transportation time and low system throughput. Route segments are shared for multiple vehicles, leading to potential congestion and deadlocks. With the advent of more powerful onboard processors and advanced sensors, it is now possible for AGVs to drive without physical guide-paths. Some experimental systems have indeed been developed [18]. Preliminary simulation results showed that free-range routing was on average 19% shorter than traditional mesh-based routing, and 53% shorter than loop-based routing. Huge potentials are therefore seen for free-range routing to improve transport capacities of AGV systems at container terminals.

Nevertheless, there are two major operational uncertainties for AGVs at container terminals, namely, dynamic task requirements, and uncertain traffic conditions. Dynamic task requirements are mainly due to the variation of vessel arrival time, and handling time of quay cranes. Uncertain traffic conditions are mainly due to stochastic interferences between AGVs [18]. Hence, a team of decentralised free-range AGVs working at a container terminal is a good test-bed to validate the proposed multi-robot motion planning approach.

The AGVs work in an area of 600m × 150m. Each AGV measures 12m × 4.5m ×1.5m and weighs 25 tonnes. The maximum velocity and the maximum acceleration of an AGV are $V_{\max} = 7\,m\,s^{-1}$ and $a_{\max} = 1\,m\,s^{-2}$, respectively.

Inertial measurement units (IMU) and sonars are used in this paper. An IMU is a device that utilises measurement systems such as gyroscopes and accelerometers to estimate the relative position, velocity, and acceleration of a vehicle in motion [3]. Sonars are common range sensors in mobile robotics. The general principle is that the system emits sound pulses and awaits the return of echoes that have bounced off from objects in the environment. Knowing the transmission speed of sound in the medium and the time of flight, it is possible to compute the distance. This method is widely used due to the low cost of sensors with adequate performance [5]. Rotational scan at the rate of 20Hz is used, with 180° field of view. The range of sonar scanning *R* is derived as follows. Consider an extreme case where two AGVs, heading directly towards each other without yaw steering, are braking from the maximum velocity $V_{\max}$ with the maximum acceleration $a_{\max}$. According to kinematic equation: $\frac{R}{2} = V_{\max} t - \frac{1}{2} a_{\max} t^2$, and $t = \frac{V_{\max}}{a_{\max}}$, *R* is derived as:

$$R = \frac{V_{\max}{}^2}{a_{\max}} \tag{3.1}$$

that is, *R*=49m, approximately four times the length of an AGV. With a proper yawing angle, this sensing range can sufficiently safeguard the motion safety. Empirical selection of the parameters of the navigation function is as follows: $K = 100$ Nm^{-1}, $A = 3.6\times10^5$ N·m, $B = 2.5\times10^4$ N, and $C = 60$ m.

Fig. 7 shows a scenario where three AGVs plan their respective motions to their goals while avoiding collisions. In Fig. 7(a), the red AGV detects two imminent neighbours: the blue AGV and the green AGV. The blue AGV exerts stronger repulsion and the composed repulsive force steers the red AGV to the right. In Fig. 7(b), while the potential collision between the red AGV and the blue AGV has been eliminated, the red AGV detects the green AGV as an imminent neighbour. Hence, the red AGV, which is repelled by the force from the green AGV, steers to the left to evade the green AGV. Note that this time, the non-imminent neighbour blue AGV has no influence on the motion of the red AGV, avoiding redundant movements of the red AGV and with less computation overhead. In Fig. 7(c), the potential collision between the red AGV and the green AGV has been avoided, and the red AGV continues to approach its destination.

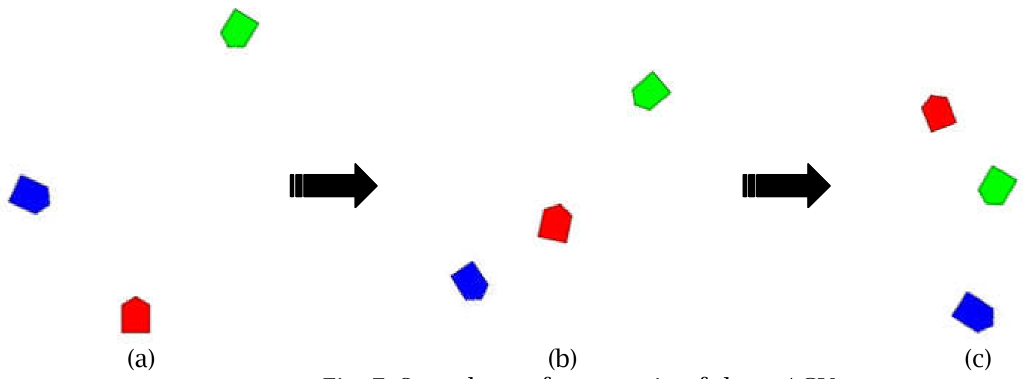


Fig. 7: Snapshots of a scenario of three AGVs.

To verify the merit of the motion planning approach with the module for detection of imminent neighbours, a simulation of ten AGVs to transport one hundred containers is carried out. Fig. 8 shows the comparison of operational times of traditional approaches without the detection module and the proposed approach with detection module. It can be observed that an improved efficiency by 16.2% is attained. It verifies that redundant computation overheads and robot movements can be avoided with the detection module, effectively streamlining the motion planning and collective performance of the team of AGVs.

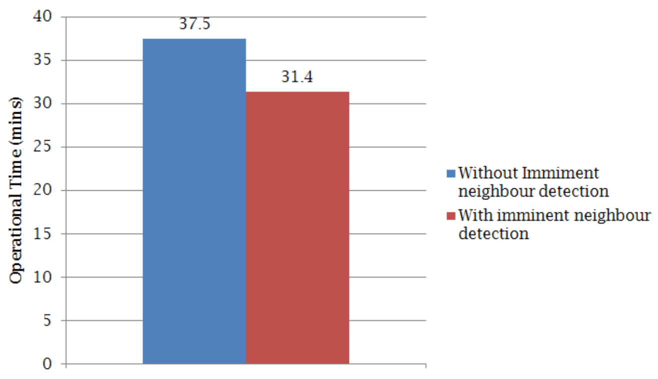


Fig. 8: Comparison of approaches with and without detection of imminent neighbours.

Two implementation issues are worthy of discussion. The first issue is the cycle processing time of the AGV team. The team of physical AGVs at a real-life container terminal are designed to be decentralised with independent and simultaneous motion planning. However, in computer simulations as demonstrated above, AGVs are inevitably processed sequentially by the computer program. Hence, the more AGVs involved in the team, the longer it takes to process and update all the AGVs. This would lead to a potentially dangerous situation where some AGVs that have not yet been updated with current information may collide with obstacles. Indeed, to simulate a large number of AGVs, the computer program design for the behaviours of each AGV and the program control logic should be optimised to render its execution as efficient as possible.

Another issue is the sensing noises of a sensor. Both the inertial measurement units (IMU) and sonars used in this simulation have measurement noises. Indeed, they resemble the physical sensors in real-life applications. Hence, adequate tolerances should be incorporated accordingly during implementation to make the measurements of relevant sensors effective.

4 CONCLUSIONS AND FUTURE WORK

This paper presents a bio-inspired intelligent approach to motion planning for mobile robots in dynamic environments. It is inspired by the motion behaviours of creatures in a crowd which includes local sensing, detection of imminent neighbours, and navigation. The motion of each robot is governed by a navigation function in the form of Newton's law of motion, describing the dynamics of virtual forces on the motion of each robot. The proposed motion planning approach features a module to spot imminent neighbours, reducing computation overheads and eliminating redundant robot movements. Moreover, the calculation of repulsive forces is a two-section function, making the repulsive forces more adaptive to the degree of collision danger. The system design and the motion planning approach are validated with a simulated case study.

It should be noted that, like most of the similar motion planning approaches, the navigation parameters of the proposed approach are empirical, determined by trial and error, and there is still a lack of effective analytic guideline to design the parameters. It would be more fruitful if a preliminary analytical model could be derived. Secondly, the proposed motion planning approach works in a decentralised architecture in which each robot autonomously plans its own motion, without any intervention from a central decision-maker. Like other decentralised motion planning strategies, optimal decentralised solution for a single robot may not necessarily aggregate to an optimal global solution at the collective team level. One of the challenges of decentralised control is to predict the collective team performance based on individual decision making, and vice versa. Nevertheless, no effective and analytic model has been developed yet [9]. It would be another contribution if such a model can be established to analyse the performance of the proposed decentralised motion planning approach, and other similar ones. Thirdly, the proposed motion planning approach mainly focuses on solving the stochastic inter-vehicle collisions. It would be more versatile if the issue of task allocation could also be taken into account. Moreover, dynamic task request is ubiquitous in real-life multi-robots applications, which definitely influences the motion planning strategy. Future work will be devoted to addressing this interesting yet challenging issue.

ACKNOWLEDGEMENT

The authors would like to thank the CRCG and the HKU SPACE Research Fund of the University of Hong Kong for the financial support for this project.

REFERENCES

[1] Choset H.; Lynch, K.M.; Hutchinson, S.; Kantor, G.; Burgard, W.; Kavraki, L.E.; Thrun, S.: Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT Press, Massachusetts, USA, 2005.

[2] Cortés, J.; Martínez, S.; Karatas, T.; Bullo, F.: Coverage Control for Mobile Sensing Networks, IEEE Transactions on Robotics and Automation, 20(2), 2004, 243–255. doi:10.1109/TRA.2004.824698

[3] Dudek, G.; Jenkin, M.: Chapter 20: Inertial Sensors, GPS, and Odometry. In: Bruno, S; Oussama, K.: Springer Handbook of Robotics, Springer, The Netherlands, 2008.

[4] Helbing, D.; Buzna, L.; Johansson, A.; Werner, T.: Self-Organized Pedestrian Crowd Dynamics: Experiments, Simulations, and Design Solutions, Transportation Science, 39(1), 2005, 1-24. doi:10.1287/trsc.1040.0108

[5] Kleeman, L.; Kuc, R.: Chapter 21 Sonar Sensing. In: Bruno, S; Oussama, K.: Springer Handbook of Robotics, Springer, The Netherlands, 2008.

[6] Murray, R.M.: Recent Research in Cooperative Control of Multivehicle Systems, Journal of Dynamic Systems, Measurement and Control, 129(5), 2007, 571-583. doi:10.1115/1.2766721

[7] Pallottino, L.; Scordio, V.G.; Bicchi, A., and Frazzoli, E.: Decentralized Cooperative Policy for Conflict Resolution in Multivehicle Systems, IEEE Transactions on Robotics, 23(6), 2007, 1170-1183. doi: 10.1109/TRO.2007.909810

[8] Parker, L. E.: ALLIANCE: An Architecture for Fault Tolerant Multirobot Cooperation, IEEE Transactions on Robotics and Automation, 14(2), 1998, 220–240. doi:10.1109/70.681242

[9] Parker, L.E.: Chapter 40: Multiple Mobile Robot Systems. In: Bruno, S; Oussama, K.: Springer Handbook of Robotics, Springer, The Netherlands, 2008.

[10] Parker, L.E.: Path Planning and Motion Coordination in Multiple Mobile Robot Teams. In: Meyers, R., editor-in-chief: Encyclopedia of Complexity and System Science, Springer, The Netherlands, 2009.

[11] Peasgood, M.; Clark, C.; McPhee, J.: A Complete and Scalable Strategy for Coordinating Multiple Robots within Roadmaps, IEEE Transactions on Robotics, 24(2), 2008, 283-292. doi:10.1109/TRO.2008.918056

[12] Player/Stage Project, http://playerstage.sourceforge.net/

[13] Richards, A.; Bellingham, J.; Tillerson, M.; How, J.: Co-ordination and Control of Multiple UAVs, AIAA Conference on Guidance, Navigation, and Control, 2002, Monterey, California, USA.

[14] Shima, T.; Rasmussen, S.; Sparks, A.; and Passino, K.: Multiple Task Assignments for Cooperating Uninhabited Aerial Vehicles Using Genetic Algorithms, Computers and Operations Research, 33(11), 2006, 3252–3269. doi:10.1016/j.cor.2005.02.039

[15] Silveira, R.; Dapper F.; Prestes, E.; Nedel, L.: Natural Steering Behaviors for Virtual Pedestrians, Visual Computer, 26(9), 2009, 1183-1199. doi:10.1007/s00371-009-0399-0

[16] Stahlbock, R.; Vob, S.: Vehicle Routing Problems and Container Terminal Operations---An Update of Research. In: Golden, B. et al.: The Vehicle Routing Problem. Springer, The Netherland, 2008.

[17] Sud, A.; Gayle, R.; Andersen, E.; Guy, S; Lin, M.; Manocha, D.: Real-time Navigation of Independent Agents Using Adaptive Roadmaps, The ACM Virtual Reality Software and Technology 2007, Newport Beach, California, USA.

[18] Vis, I.F.A: Survey of Research in the Design and Control of Automated Guided Vehicle Systems, European Journal of Operational Research, 170(5), 2006, 677-709. doi:10.1016/j.ejor.2004.09.020

[19] Zavlanos, M.M.; Pappas, G.J.: Dynamic Assignment in Distributed Motion Planning with Local Coordination, IEEE Transactions on Robotics, 24(1), 2008, 232-242. doi:10.1109/TRO.2007.913992