人工知能学会
第15回 SIG-Challenge研究会

2002

RoboCup

2002年３月22日
日本科学未来館
（2002年 ロボカップ春季競技会）

目次

Can the target of RoboCup be attained ?

Masahiro Kosaka
HEW(**H**umanoids for **E**very **W**here) Research
mkosaka@dab.hi-ho.ne.jp

Abstract: Can the aggressive target which RoboCup referred to as {Making the robot team which can Win human's World Cup championship team by 2050} holds up be attained?. This possibility was examined from two viewpoints. One is the development speed of the artificial intelligence and the realization possibility of artificial intelligence equal to a human intelligence. Secondly a possibility that a soccer robot could be more realized using the main devices which constitute a robot was examined. It was concluded that the target which RoboCup holds up could sufficiently be attained. Even if it did not expect the emergence of the material or a device which is not yet seen, like a dream, it was predicted that the humanoid having the capability about the same as human was realizable by 2050 using the latest technology of the time. This humanoid penetrates to corner of society and will contribute greatly to realization of ideal society and happiness of mankind.

1 Introduction

The target which RoboCup referred to as "making the robot team which can {Win human's World Cup championship team with human's rule by 2050} holds up is considered to be an extremely difficult target. It is very interesting subject whether can this magnanimous target attain sure enough or not. In order to make such a soccer team by the robot, it is necessary to realize the humanoid type robot which constitutes the team. The outstanding capability than most people in physical strength, cognition and judgment capability is required for a soccer robot. It is understood that even realization of the robot having the capability of the usual average human is very difficult. Enormous difficulty will be expected in realization of the robot with the capability which exceeds human's capability. Is it realizable sure enough from now on within 50 years?. This paper examined this possibility from two viewpoints. One is the development speed of artificial intelligence by the computer, and the realization possibility of artificial intelligence having the same capability of ordinary human. Secondly the main devices which constitute a robot examined whether it was the level which can realize a soccer robot. Consequently, it was concluded that the target which RoboCup holds up could sufficiently be attained.

2 Importance of clear and concrete target

The RoboCup initiative involves in the persons concerned in the world, and becomes as big movements. The clear and grand target said that a robot challenges and wins to man's World Cup champion is held up in 2050. [1]-[8] The present engineering level is far low as compared with the target. Also in RoboCup, engineering level of vision recognition of a ball / player / field is quite inadequate. Although vision recognition research has a long history, it has stopped at such a low level. It is imagined that one of these causes is that there was no setup of a clear and concrete target in research and development. Although it is thought that difficulty follows on vision recognition in the environment where a situation changes continuously, it is clear to the actual robot that it cannot be used if it is not the technology corresponding to such environment. It will be a reason why the clear and concrete target are required in the research and development. RoboCup initiative is the suitable challenge field in which various component engineering can vie to each other. The challenge which sets up a clear and high target greatly accelerates technical development. The U.S. Apollo Project is proving this clearly. Much more people are wanted to participate in these initiatives, to muster the wisdom in the world, and to desire promotion of a robot's technical development.

3 Technologies required for soccer robot

The key issue of soccer robot realization is the development of "humanoid type robot" with the body and intellect equivalent to human. Various technology, devices and material are needed for the realization. The main thing is listed below.

- Vision cognitive capability that the situation of the field can be grasped on real time.
- Derive the optimum solution of action which should be taken to the next from a position and a motion of the ally-and-oponent player and a ball in the field .
- Derivation of the control command to each part of the body of operation based on the optimum solution.
- The tough body which it can runs or operate a ball by foot based on a commander of operation.
- The small lightweight energy source supporting a long and extreme motion.

That is, muster of the latest and wide-range technologies are needed for realization of a soccer robot. They are structure material, portable energy source, various sensors, an actuators and servo control and a computer hardware/software, artificial intelligence, and knowledge engineering and so on. These component technologies are not peculiar to a robot, and are used in many other fields. Therefore, it is possible to utilize effectively for a robot many component technologies which carried out progress development in other fields. A computer and

artificial-intelligence technology are the example of a type. Moreover, it is used in fields other than a robot, and the component technologies which is conversely developed in a robot field and progressed can also be contributed to development of the concerned field. The application possibility to a robot is examined about such technology below.

4 Emergence of physical capability

Physical abilities and mental abilities are very closely related, and arguing about both independently will not be proper. Although the force and movement speed which each joint generates are the one aspect of physical abilities, it will be far from sufficient to realize body movement. In order to walk along a road, for example, while avoiding a passerby and an obstacle, recognition and understanding of surroundings are inevitably needed at first place. Furthermore, derivation of body movement by grasping one's present situation is advanced mental abilities. Walking by two feet or grasping objects by hands are generally understood that it is carried out by a robot's physical capabilities. But ,in reality, ability is that the most is based on knowledge and experience. For example, exact recognition of the character of the material of the place and a level difference of the ground is executed by not mere vision capability but advanced judgment capability based on knowledge, and experience of machine. In order to continue a walk, what posture should be carried out and how to take down a leg is judged 100% based on knowledge of robots. Therefore, it can be said that a walk is knowledge and is wisdom. It is a premise, of course, to have firm and strong frame, a forcible actuator, and exact and flexible control capability. In order for a robot to derive required correspondence such as the recognition capability, the handling capability of the subject by fingers, and safe and versatile locomotion capability in various environment, comprehension capability of various information, and to perform it correctly, it is necessary to have the mental abilities about the same as human. It is the same also in sensing, in vision and in hearing capability. It is only possible to see and hear by excellent ability of the brain. Although it is possible for the primitive sensing organ, to detect sound pressure, light intensity and a pattern of light, it is not possible to comprehend speech or visual pattern. It is indispensable to utilize an advanced brains capability.

5 Form of robot body

5.1 Frame of robot body

In order to constitute a robot's frame, a light and tough material like human's bone is demanded. For this purpose, use of the various composite materials which showed remarkable development in recent years is more desirable than a

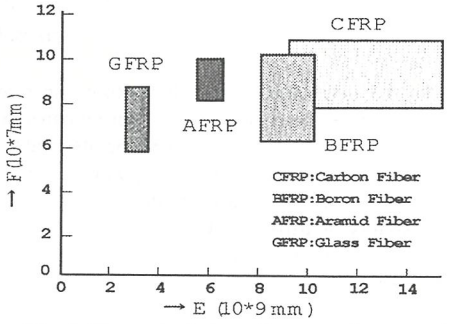


Fig-1 Characteristics of composite material

single metal and/or single plastics. These are roughly categorized to a metal composite material (MMC), fiber reinforced plastics (FRP), and a ceramic composite material. Since a metal composite material was expensive, it was only being used in a space field or a military field. However, it came to be used for automobile engine, the gas turbine, etc. in recent years due to it's cost reduction. In a robot field, probably there is no requirement for super high durability to heat, and thus the use of this material may be the subject of the future from the reason of economical efficiency. Fiber reinforced plastics (FRP) is the composite material which being made plastics as the mother material and reinforced with various fibers. There are various kinds in FRP category. They are (1) Strengthened with Glass Fiber GFRP(Glass Fiber Reinforced Plastics) (2) Strengthened with Carbon Fiber CFRP(Carbon Fiber Reinforced Plastics) (3) Strengthened with Aramid Fiber AFRP (Aramid Fiber Reinforced Plastics) . These FRPs are the lightweight and high tension structure material. Due to the wide range of characteristic and price, FRP is used in large quantities in the wide range application area. The use of FRP has attained to very large ranges, such as the space, aviation, automobile, electric parts, engineering works, construction, and sport goods. Although it must be careful of that there is heterotrpicity and relatively law heat-resistant temperature of about 300 degrees C , it is considered the most suitable structure material for the robot field. Although the ceramic composite material is used for space and medical components used for the living body, it has also have many unsolved area and to expect them to development of future research. Therefore, the use to a robot field is a future subject.

5.2 Actuators

Actuator	Features
Electromagnetic	Motor,Solenoid Wide variation
Electrostatic	Ultramicrosize Suitable for micromachine
Piezo/Magnetostrictive	Piezo,Magnetostrictive Small displacement
Hydraulic	Oil cylinder High power need oil pump,oil leakage
Pneumatic	Air cylinder Ultraclean low controlability
Miscellaneous	Shape memory,Ultrasonic motor

Fig-2 Types of actuator

The actuator equivalent to human's muscles is a device indispensable to posture maintenance of a robot, a walk, grasp of an object, movement, etc. Moreover, in order to enable various operation, very many numbers are used. The list was shown in Fig-2 with the outline about the main existing actuators. It has the respectively different characteristic and it is necessary to perform evaluation from the viewpoint used for a robot. The result of five stage evaluation of various actuators was shown in Fig-3 from this viewpoint.

Evaluation	Electromagnetic	Electrostatic	Piezo,Magnetostrictive	Fluid Motor	Pneumatic
Power Density	+2	+1	+1	+2	+2
Efficiency	+2	0	0	+1	+1
Controlability	+2	+2	+2	+1	+1
Output versatility	+2	-1	-2	-1	-1
Selection	+2	-2	-2	-1	-1
Usability/Cost	+2	+2	+2	-2	-1
Life/Reliability	+2	+1	+2	0	0
Total	**+14**	**+3**	**+3**	**0**	**+1**

+2:Excellent +1:Good 0:Marginal -1:Bad -2:Inferior

Fig-3 Evaluation result

Each evaluating point was added up simply and it resulted as comprehensive evaluation. The electric motor using electromagnetic force gained highest evaluation. This matches actual feeling and considers appropriate evaluation. An electric motor has the extensively wide choice of output power from very small output of a micromotor to a gigantic output, and has a feature of high energy efficiency as well. It has also very wide choice of external appearance form pencil shape to disc shape. Moreover, highly precise and flexible control is possible by the electric signal. Moreover, there is no concern of an oil leakage and is consequently very clean. The movement direction is also easily changeable from rotation to linear, and not only the movement direction but torque and speed can be chosen freely by selection of a gear train further. If the output power range required of a robot's actuator is taken into consideration, it will be possible to realize the most by this electric motor, and it will be considered to be the most excellent means. Although there exist opinions which expects an appearance of new actuators such as artificial muscles, it is thought that an appearance of the device which has the feature compete to an electric motor is not easy. Through appreciation of the outstanding feature of the electric motor, it important to promote realization of a robot using this means as a robot actuator. Other various actuators are considered to be used on a required aspect of affairs taking advantage of each feature.

5.3 Energy source

The consumption energy of bipedal walk of human and a walk of an animal were investigated, and the consumption energy to various working operation of human was investigated further. Moreover, the robot battery weight required for operation continuously for 1 hour using various batteries is shown in Fig-4 .[9][10] Battery weight of the robot weighted 30kg required to walk continuously for 1 hour is calculated about the kind of various batteries. If it is the most efficient lithium-ion battery, it will be good enough by the battery of 1kg or less. Although there is the room of an argument what percent of body weight is allowed for battery , if it is assumed to be 10 - 20% (about 40% of man's weight is skeletal muscle), it can walk continuously over 3 - 6 hours, and will be enough for the usual purpose.

Body Weight	Job	Energy Consum kJoul/hour	Weigh for 1 hour operatin(kg)			
			Lead Acid	NiCd	NiMH	Li-ion
Adult 60kg	Walking	756	6	4.7	3.5	1.8
↑	Ironing	576	4.6	3.6	2.7	1.3
Robot 30kg	Walking	378	3	2.3	1.7	0.87
↑	Ironing	288	2.3	1.8	1.3	0.67

Fig-4 Battery weight for 1 hour operation

If the computer equivalent to human's brains is assumed to consume energy equal to that of actuators, the duration hours by the same battery will be reduced by half. A robot works in many cases by stopping from time to time while staying in a fixed place not only in working by always moving. In this case, it is also possible to gain electric energy from outside such as a wall outlet. It is possible to save consumption of a battery and to charge, and it is also possible to extend practical duration hours considerably. It is, consequently, quite possible even to operate a robot with using the latest rechargeable battery of the time. The argument in which it is not possible to use humanoid type robot practically if a novel energy source is not developed, is understood to be not right direction.

5.4 Sensors

In order for a robot to cooperate and to work with human , it is necessary to have a sensing device similar to the various sense organs which human has. When the cup etc. is held by hands and it makes it move, the fundamental capability needed first is recognition of a

cup. By cooperation of various recognition technologies such as shape and a color recognized by vision sensor, a tactile sensing, if required, and with reference to memory, it is necessary to recognize object correctly. It is also possible to reach the same cognition result by using a sensing organ completely different from human. For example, it is also possible to recognize a cup by touching only using a tactile sensor. However recognition process will be completely differs from aforesaid recognition process. Human usually performs recognition for which it mainly depended on vision, and is not used to recognizing a cup only tactually. Therefore, it is expected that teaching a robot the recognition process is with great difficulty. If the importance of installing human's wisdom into a robot is considered, and instruction efficiently, it would be very important to use sensing devices similar to those of human as much as possible. Fortunately an artificial sensing devices have reached the level which can execute almost all human's sensing organs with few exception. There are many sensing devices showing much higher performance than human. The microphone which can detect sound which human's ear does not hear, and the image sensor which is visible even in the dark place where human is not possible to see, are the examples. There are some kind of devices which must wait to future development and improvement, such as a plane tactile and temperature sensing device which is distributed over the whole body corresponding to the skin of the human. If the various feeling devices in which the present use is possible are used, It will be, therefore, understood that sensor devices required for a robot is realizable safely upon using sensors based on the state-of-the-art technologies.

6 Emergence of artificial intelligence

6.1 Computers

The computer of robot can be realized by using newest architecture and CPU, memory tip of the time. The computer used for a robot will not be something different from a usual computer fundamentally. It is a common practice to use the latest computer technology of a time. However, since massive calculation capability is required of objective recognition by vision and control of hand and foot in the case of a robot, distributed-processing architecture which processes those processing by independent separate computer may also be used. Since a real-time operation will be the requisite, original robot OS may be used. Usual computer technology is effectively utilizable also about this OS. Since large knowledge processing is needed in addition to recognition or control of hand and foot, it is thought that the language currently used in the artificial-intelligence research field can be also effectively used. Moreover, CPU architecture which specialized in the processing knowledge may also be used. Walking by two feet or grasping objects by hands are generally understood that it is carried out by a robot's physical capabilities. But ,in reality, ability is that the most is based on knowledge and experience. For example, exact recognition of the character of the material of the place and a level difference of the ground is executed by not mere vision capability but advanced judgment capability based on knowledge, and experience of machine. In order to continue a walk, what posture should be carried out and how to take down a leg is judged 100% based on knowledge of robots. Therefore, it can be said that a walk is knowledge and is wisdom. It is a premise, of course, to have firm and strong frame, a forcible actuator, and exact and flexible control capability. In order for a robot to derive required correspondence such as the recognition capability, the handling capability of the subject by fingers, and safe and versatile locomotion capability in various environment, comprehension capability of various information, and to perform it correctly, it is necessary to have the mental abilities about the same as human. It is the same also in sensing, in vision and in hearing capability. It is only possible to see and hear by excellent ability of the brain. Although the details about a possibility that a computer will acquire the capability about the same as human will be discussed in section 8 of this report, the realization time is predicted in general to be 2020 - 2040 year. The time when the computer of the price becomes the capability about the same as human was predicted under supposition that the price of the computer which can be carried in a robot is assumed to be 1,000 dollars. Therefore, it may be far earlier than this that a more expensive supercomputer becomes the capability about the same as human .

6.2 Acquisition of knowledge

It is called a blockhead if a robot does not have wisdom. There is it not to mention useful for human society but quite dangerous existence. Therefore, in order to enable behaving usefully, it is the serious subject of future robot research how a robot is made to acquire wisdom. There are several means shown in Fig-5 for acquiring wisdom by robot .Among them, human has been using the means 1 and 2 over many years and accumulating experience very abundantly. However, this method has low efficiency and is required also for prolonged patience. The efficiency of knowledge acquisition is not only low, but one's knowledge being acquired and accumulated by years of efforts is facing the serious problem of disappearing with the man's death. Although circulation of written knowledge has been drastically improved by invention of the printing technology of Gutenberg, the efficiency of succession was very low and the generation which follows the next needed to be redone from very beginning which is zero. On the other hand, a robot can use the means 3 and 4 effectively, and has the

conspicuous feature in that the knowledge can be poured into another robot at high speed through proper communication channel. However detailed argument of knowledge processing such as how to store and handle knowledge by machine and a knowledge computer are left to the specialist of the field, the following correspondence being required of a robot, for enabling such knowledge acquisition and circulation.

1) A robot's frame structure, architecture and language of a computer must be standardized.

2) Mechanical structure and logic composition of a robot are similar with human.

The request of the first term shows that a transplant and circulation of the knowledge between the robots with different body composition or logic composition are difficult. The knowledge or the skill ,for example, for overcomes the level difference of floor which acquired by bipedal walking robot is hardly useful to the robot of wheel movement obviously. Also in knowledge processing, it is the same and the first requirement is indispensable to mutual practical use of knowledge. It is easily imagined

Method	Contents
1.Teaching	Teaching by human or robot who have acquired knowledge already or knowledge server.
2.Learning	Acquire knowledge by self learning mimicry, trial/error, reinforced learning.
3.Instill and store	Inject knowledge via I/F from human or robot who have acquired knowledge already and store in the robot brain.
4.Instill momentary	Acquire knowledge via I/F from human or robot who have acquired knowledge already or knowledge server but not store in the robot brain.

Fig-5 Acquisition of knowledge

to be quite difficult in transplanting application software between computers with different OS or a language. Use of emulation technology is possible, and it is imperfect and also takes time. Therefore, the application of emulation technology may be limited. Realization of the robot with capability equivalent to human is considered as an actual problem by rapid progress of a computer technologies. However, it must be only potential capability, and finally grow up to a humanoid which can demonstrate the capability level with human after acquisition of various knowledge like cognitive capability , judgment capability, and various athletic abilities . It is called a BLOCKHEAD if a robot does not have wisdom. There is it not to mention useful for human society neither quite dangerous existence. Therefore, in order to enable behaving usefully, it is the serious subject of future robot research how a robot is made to acquire wisdom. It is indisipensably important to teach robot and to learn by himself. The knowledge acquired by one can be poured into another robot at high speed through proper communication channel. The technique of knowledge acquisition of a robot will be reformed revolutionary and it will supercede over knowledge acquisition of human. The second request is conditions important when man teaches a robot , or a robot learns himself by mimicry for acquiring knowledge. Although situation will be different if it becomes the phase where a robot can acquire knowledge by himself without help of human, learning from human will be most powerful means for acquiring knowledge at least for the time being. This requirement is indispensable, when instill new wisdom into a robot or skill is taught, and when a robot imitates operation of human is to learn and gain new action or skill. It would be difficult or almost impossible for instilling knowledge into the robot with which logical system and a thinking system are completely different from human. Similarly it is very difficult to instill the skill which human has to the robot with which body system is totally different. How it is possible to instill the skill which gained by manipulating five fingers to the robot of three fingers?. Since the knowledge stored in the robot goes back and forth across time and space, a robot's knowledge is expanded quickly, and as a consequence it will be a matter of time when total knowledge of the robot exceeding that of human!. Furthermore, even if the knowledge which one robot holds is restricted by the storage capacity or operation capability, support of the enormous amount of knowledge can be obtained through a network from another robots and/or servers out of a robot. That is, if the global networks which are quickly developed recently, such as the Internet, are utilized, it is possible to utilize even global knowledge and the total amount of the wisdom which a robot can utilize substantially is infinite. Thus, since knowledge can circulate across time and space, the intelligence which was excellent in all ages and countries can be utilized freely. It expands even quickly ignited by birth of the new existence called robot in this way, and the wisdom will show explosion, which is called "Big Bang of Wisdom!" .

7 When soccer robot appears ?

The robot drawn by Karel Capek in 1920 is going to appear to us. An appearance of the humanoid robot which has capability similar to human is not in the dream. It is possible to predict the appearance of robot from speed of development of the computer technology which is supported by microelectronics. An appearance of a robot is not the fairy tale of the future but the talk of the surprisingly near future. It would be quite natural to predicts the appearance time of the robot with the capability about the same as man from development of a computer and artificial-intelligence technology. By wonderful progress of microelectronics known for the law of Moore, the improvement in computer capability and the fall of price are progressing rapidly. As a result, the calculation power per unit cost is improving at surprisingly high speed. H. Moravec and R. Kurzweil

had performed investigation of the calculation power per 1,000 dollars which is the price considered to be permitted as a robot's computer, and future prediction to the humanoid robot. Moreover, using the result of human's nerve physiology, the estimation of human's amount of memory and logic operation power were performed. The comparison with a computer to human was discussed.[11]-[14]
The short-term prediction to the year 2020 is shown in Fig-6.

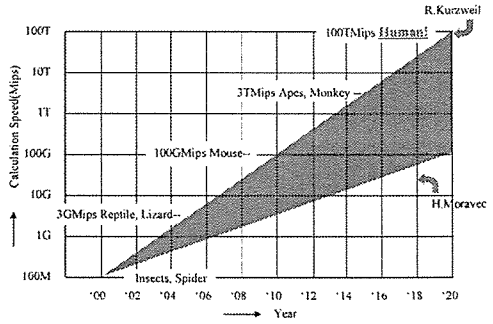


Fig-6 Forecast of Computing Power (Shot term)

If it considers that ability of various kinds of body operation and recognition, is governed by brains capability, the robot having the brains about the same as human will be appropriate to predicting that the recognition capability and physical ability about the same as human. Kurzweil and Moravec predicted that it comes to have capability of human at year 2020 and year 2040 ,each respectively. The target referred to as that robot soccer will challenge the World Cup champion in 2050 has enough validity, and it is thought that it will be attained.
If it is simple work, a robot of having ability less than human can work at our home. The humanoid robot will penetrate to society early from the above-mentioned prediction. Moreover, use of the computer which exceeds 1,000 dollars in the early stages of introduction should also be permitted, and early introduction is predicted also. It should be interpreted that the time when robot is equipped with the capability about the same as human as the time which robot is playing an active part in society in large quantities. It is predicted as that to which a certain amount of market has risen in year 2010. The time of about 50 years passed from IBM701 computer in 1952, 24 years after sale of Apple II, 30 years after Intel 4004 which is the first microchip. When the development speed of the latest computer is considered, I can never think optimistic prediction of year 2020 which Kurzweil predicts. Taking into consideration of the factors already described, robot's introduction will be much earlier than generally expected. The humanoids robot is approaching to just next corner is not an overstatement.

8 Conclusion

This possibility was examined from two viewpoints. One is the development speed of the artificial intelligence and the realization possibility of artificial intelligence equal to a human intelligence. A possibility that a soccer robot could be more realized using the main devices which constitute a robot was examined. It was concluded that the target which RoboCup holds up could sufficiently be attained. Realization of the soccer robot is realizing the robot which is equal to realize human symbiosis robot as well. Realization of a ubiquitous robot coexisting with human in every corner of our society, and plays an important role. Moreover, this robot carries out a huge contribution to our society. Therefore, RoboCup is the great challenge to realization of human beings' dream. Target achievement is possible if it is tackled with our best effort. I personally believed in the contribution of [RoboCup Project] to the happiness of mankind will far greater than that of [Apollo Project].[15]

References

[1] H.Kitano(Ed.): RoboCup-97: Robot Soccer World Cup1,Lecture Note in Artificial Intelligence 1395,Springer (1998)
[2] M.Asada ,H.Kitano(Eds.): RoboCup-98: Robot Soccer World Cup2, Lecture Note in Artificial Intelligence 1604,Springer (1999)
[3] M.Veloso,E.Pagello,H.Kitano(Eds): RoboCup-99: Robot Soccer World Cup3, Lecture Note in Artificial Intelligence 1856,Springer (2000)
[4] P.Stone,T.Balch,G.K.Kratzschmar(Eds): RoboCup-00: Robot Soccer World Cup4, Lecture Note in Artificial Intelligence 2019,Springer (2001)
[5] H.Kitano,M.Asada: An Overview: RoboCup, today and tomorrow,IROS'99, Tutorial, TS4(1999)
[6] M.Asada,S.Suzuki,M.Veloso,G.K.Kraetzschmar,and H.Kitano:What We Learned from RoboCup-97 and RoboCup-98,pp1426-1431,Proc.1999 IEEE/RSJ Int'l.Conf. on Intelligent Robots and Systems(1999)
[7] H.Kitano,M.Asada:RoboCup Humanoid Chllenge:That's One Small Step for A Robot,One Giant Leap for Mankind,Proc.IROS-98(1998)
[8] H.Kitano,M.Asada,I.Noda,H.Matsubara: RoboCup:Robot World Cup,pp30-36,IEEE Robotics & Automation Magazine(Sept.1998)
[9] David A.Winter:Biomechanics and Motor Control of Human Movement,pp103-140,John Wiley & Sons(1990)
[10]P.M.Alexander:Exploring Biomechanics:Animals in Motion,W.H.Freeman(1992)
[11] H.Moravec: Robot, Oxford Univ.Press,pp91-126(1999)
[12] R.Kurzweil: The Age of Intelligent Machines, MIT Press,pp425-458(1992)
[13] R.Kurzweil: The Spiritual Machines, Penguin Books,pp101-132(1999)
[14] H.Moravec: MIND Children, Harvard Univ.Press,pp51-74(1988)
[15] M.Kosaka:Humanoids in Every Home,Proc.Humanoids2001,pp101-132(2001)

RoboCup の向こうに何が見えるか？：その光と影
What We can Envisage over the Horizon of RoboCup ?

小坂 雅博
Masahiro KOSAKA
HEW(Humanoids for Every Where) Research
mkosaka@dab.hi-ho.ne.jp

Abstract

[By the year 2050, develop a team of fully autonomous humanoid robots that can win against the human world soccer champions] Can the aggressive target which RoboCup holds up be attained? .This possibility was examined from two viewpoints. One is the realization possibility of the artificial intelligence similar to human. Secondly a possibility of realizing a soccer robot by using main devices which constitute a robot was examined. Consequently, it was concluded that the target which RoboCup holds up could sufficiently be attained. Without expecting the appearance of the material like a dream, or a device which is not yet seen, it was predicted that the humanoid having the capability about the same as man was realizable by 2050 using the latest technology of the time. The humanoid advances to every corner of society and will contribute the happiness of human being and realization of the ideal society.

1 はじめに

１９９７年名古屋での第一回大会以来 RoboCup プロジュエクトは年々発展を遂げ、世界の関係者が数多くの参加する、世界的な活動になっている。RoboCup が掲げる壮大な目標は、それが達成された暁には、光り輝く理想社会の実現が展望されている。本論文では RoboCup の意義を振返り、光と影を分析し、将来へ向けての提言をまとめた。

	Yaer	City(Country)
No1	1997	Nagoya(Japan)
No2	1998	Paris(France)
No3	1999	Stockholm(Sweden)
No4	2000	Melbourne(Australia)
No5	2001	Seattle(America)
No6	2002	Fukuoa/Pusan(Japan/Korea)

Fig-1 History of RoboCup

2．RoboCup の意義

RoboCup は技術発展を研究面から促進する従来の【学会活動】や、競い合う楽しさを通じて青少年に科学技術に対する興味を喚起する【ロボットコンテスト】の何れとも異なったユニークな活動である。〝２０５０年までにワールドカップチャンピオンチームに勝つロボットチームを作る〟事を活動の目標として掲げ、心身とも優れた人間型ロボット（ヒューマノイド）の実現を標榜している。人間社会と共存して活動する、いわゆる『ユビキタスロボット』の実現にもっとも効果的かつ効率的に活動と考えらる。

2．１目的の明確な活動

要素技術レベルは、その目標に比較して遥かに低いものである。ロボカップにおいてボール/プレイヤー/フィールドの視覚認識は、まだまだ不十分極まりない状況である。これだけ長期間の視覚認識の歴史を持ちながら、このような低レベルに留まっているのは研究開発に具体目標の設定がなかったのではないかと想像する。絶えず状況が変化する環境での視覚認識には困難が伴なうと思われるが、その様な環境に対応する技術でなければ、現実のロボットには使用できないことは明らかである。要素技術ばかりでなく、明確な具体的目標を定めた研究開発が必要な所以である。明確で高い目標を設定してのチャレンジは、米国のアポロ計画を見るまでもなく、技術開発を大いに加速することは歴史が証明している。RoboCup は『 2050 年までに人間のワールドカップ優勝チームに、勝つロボットチームを作る』と言う明確な目標を掲げており、その目標の達成経過を明確に評価する事が可能である。

2．２テクノロジー指向からアプリケーション指向へ

殆どの研究開発は本来、その研究が何のために使用されるのか明確に想定されている必要がある。すなわちアプリケーション指向でなければならない。応用を明確にする事によって研究の方向づけや進捗の客観的な評価が可能になる。これによって“木に登って月に近づいた”と言う愚はを避ける事ができる。

特定の要素技術の種類に囚われることなく、目的達成に最適な手段が選択される。例えばフィールド内のサッカーボールの認識を目標にするのであれば、その為の最善の要素技術を選択すれば良く、必ずしもビデオカメラによる視覚認識手段に拘る必要はない。超音波やレーザービームによる測距技術を活用したり、或いは触覚さえも組み合わせる事によって最も正確、かつ効率的に目的を達成する事も可能である。巧まずして“センサー融合”が実現される事になる。また物体認識などにおいて対象物を様々な手段で観察するとともに、その同定に於いてはロボットの持つ知識を活用される。すなわちさらに一歩進んで“知識融合”が実現される事になる。極端な場合は物体に有線／無線の認識タグを取りつけ、それを読取る事で対象物の同定ができれば最も簡単で確実に目的を達成できる事になる。すなわち目的の明確な研究は手段の多様性をもたらす事になる

Research Orientation		
Technology Driven	→	Application Driven
Seeds Driven	→	Needs Driven
Research Driven	→	Market Driven

Fig-2 Application Oriented Research

2．3競争と公開

研究にコンペティションは馴染まない、との意見もあるが、公正で公平な競争は疑い無く強いモチベーションになる。研究成果が共通の土俵で競争し、明確な形で評価される事は研究を効率的に推進する原動力になる。RoboCup はその土俵がサッカーフィールドであり、そこで磨かれ評価されるロボットは、社会に進出するロボットの実現に直接つながるものである。また RoboCup ではシンポジュームを同時に開催し、技術が公開される仕組みになっており、技術交流と相互刺激が促進されることも重要な事である。閉鎖的で一人よがりな研究からは真に力強い技術は生まれ難いと考える。

Features	
1	Setup Clear and Aggressive Target
2	Seek for not Seeds but Needs Oriented Research
3	Open and Competitive Promotion

Fig-3 Features of RoboCup Project

3 RoboCup はロボット実用化への近道

RoboCup の目的の一つは人間と共存し、社会に役立つロボットを実現する事である。サッカーと言う実社会に近い環境で活動するロボットの研究開発は、社会で活動を前提とするロボットの実現に直接貢献するものである。“月へ人間を送り込む”アポロ計画での成果は間接的にしか実社会に貢献しないのに比べ、“ワールドカップ優勝チームに、勝つロボットチームを作る”と言う RoboCup 計画での成果は人類への直接的な貢献が可能である。さらにロボットの素材、エネルギー源、コンピュータ、人工知能・・など幅広い技術分野の集積であり、この推進を通じて獲得された技術は極めて幅広い波及効果がある。

Project	RoboCup	Apollo
Features	◎ Robot soccer player can work at our society. ◎ Repercussion effect can be enormous.	× Man reach to the Moon can not help our life directly. ○ Repercussion effect can be large.
Contribution	Direct and Enormous	Indirect and Large

Fig-4 Contribution to Mankind

4 機械要素支配から知能要素支配へ

ボールをドリブルしながらフィールドを走り回ったり、相手のすきを突いてゴールへボールをシュートするためにはロボット選手に対して、身体にも増して優れた知能が要求される。初期のロボットは障害物のない平坦な場所でしか移動できない、いわゆる【原始歩行】の能力であった。ここでは頑丈な骨格、強力なアクチュエータやサーボ制御技術など機械要素が支配的であった。極端な場合はモーターとカム、リンクのみで歩行を実現した例もある。また緩やかな坂を歩いて下るパッシブ歩行や、身体を揺すりながら垂直の棒を下りるおもちゃの鳥などもあった。ここでは周辺の情況を観察、認識したり、自分の姿勢を正しく認識して、身体をどのように動かすかの知能的要素は希薄であった。しかしながらサッカーロボットは常に周辺の情況や自分の姿勢などを認識し、最も望ましい行動を的確に判断する高度な知能が要求される。すなわち RoboCup のロボットは機械要素より圧倒的に知能要素に支配されて制御されなければならない。人間と共存するユビキタスロボットは、その行動を主として支配するのは知能要素であり、機械要素は従である。初期のロボットは機械要素／身体能力が支配的であるが、次第に様々な情況

を認識する感覚能力が強く支配するようになり、最終的には知能が殆どの支配を行う事になる。このようにロボットの発展段階を考慮してもRoboCupのアプローチは正鵠を突いたものであろう。

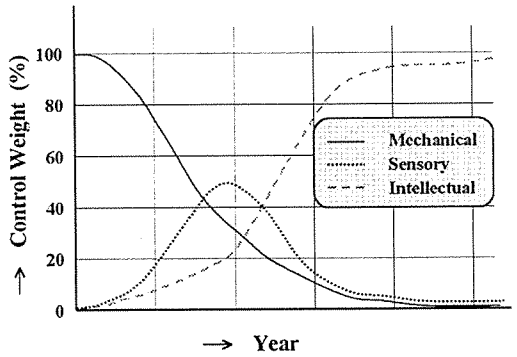


Fig-5 Change of Ruling Power

5 ロボット研究と人工知能研究の連携

ロボット研究の目的の一つは先に述べたように人間並みの能力を持ったロボットを実現する事である。RoboCupの目標も心身ともに優れたサッカーロボットを作る事であり、目的は共通である。このロボットは身体能力にも増して優れた知能を備える必要があり、ロボット研究は人工知能／人工知識の研究に逢着する。

知識を持たないロボットはまさに〝木偶の坊〟であり、人間にとって有用どころか、危険極まりない存在である。知能がなければ足があっても歩く事もできず、目や耳を持ち光や音を感知できても何も見えづ、聞こえない事になる。原始的な身体の動きや感覚を除き、殆どの感覚や運動はすべて知識／知能によって支配されている。すなわちロボットの研究は人工知能研究そのものである。

【知能を持った手足】ではなく【手足を持った知能】の実現を目指さなければならない。

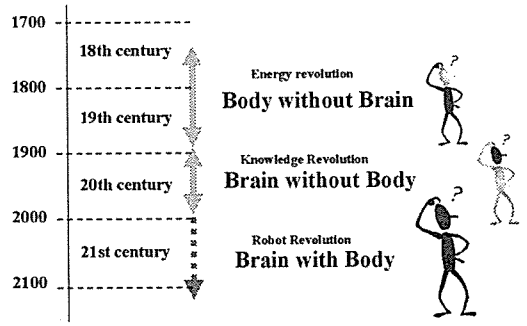


Fig-6 Brain with Body

一方現実は［ロボット研究］と［人工知能研究］は乖離していると言わざるを得ない情況である。両者の関係は希薄であり、今後思い切った連携、協力が必要である。内外の両学会の論文テーマの分析・比較からもこの現状は明らかである。ロボットが〝木偶の坊〟にならないためにも、両者の十分な協力、さらに進んで融合が焦眉に課題である。両者の協力が必要な舞台は〝ユビキタスロボット〟であり、ロボットは人工知能研究の支援を得て本当の【手足を持った知能】になり、人工知能研究はロボットと言う活動の舞台を得て【ＡＩ飛躍の時代】をようやく迎える事になる。RoboCupは両者を結び付ける絶好の舞台であろう。

Field		Main Activity
Robotics Research	--	Mechanical, Control and Application with slight AI
AI Research	--	Knowledge, Language and Cognition with slight Mechanics

Fig-7 Main Field of Robotics and AI Research

6 学会と産業界の乖離

1960年代にスタートした工業用ロボットの時代はロボット研究の学会とその実用化をになう産業界は良好な関係を維持していた。しかしながらその後両者の関係は疎遠になって久しく、最近のロボット関係学会への産業界からの論文発表や出席参加は極めて少ないのが現状である。この情況を憂い警鐘を鳴らす人もいるが大きな動きにはならづ、情況の改善は進んでいない。理由は双方がその必要性を痛感していない事であろう。

Side		Observation by another side
Academy	--	Industry always ask for instant remedy and does not comprehend needs of long term research Industry always ask for instant remedy and does not comprehend needs of long term research
Industry	--	Academy can not supply tomorrow business we have to help ourselves Academy fails to show future vision and road map of robotics

Fig-8 Academy and Industry

このような情況が続くと両者の間に無用な相互不信が生ずる事が懸念される。早急な改善が望まれるところである。産業界が比較的短期的な収益を希求し、長期的な展望に欠ける事は洋の東西を問わず一般的である。この打開にはロボットによってどの様な世界が拓けてくるかの【将来ビジョン】、その実現へ向かっての【ロードマップ】を策定し、これを明らかにする事が重要である。先ずこれをアカデミズム側が作成し工業界側へ提示する必要がある。これに対しアカデミズム側が躊躇したり、甚だしい場合は忌諱している傾向がある。遥か将来の将来の夢物語ではなく今からせいぜい10年、15年と言う近未来の話しである。この【将来ビジョン】および【ロードマップ】が明確でない事が学会と産業界の乖離を惹起している最大の原因と思われる。RoboCupはこの点においても壮大なビジョンを掲げて推進されている事は、上記問題に鑑みて特筆大書すべき事である。

7 RoboCupの光と影

人類への未曾有の貢献が期待されている〝人間共生ロボット〟の実現に最も直接的な寄与が期待され、さらにその実現には極めて幅広い技術分野の糾合が必要となるテーマを最も効果的な方法論で推進しているRoboCup計画はまさに希有なプロジェクトである。よくアポロ計画と比較されるが、人類への貢献の大きさにおいて遥かに上を行く“Super Big Project”である。これをRoboCupの光の面とすれば、一方それに付随してRoboCupの影の面が存在している。以下この点に付いて検討したい。最も重大な問題は“計画達成に必要な十分な研究資源の投入が無い”と言う事である。先にも述べたように目標の達成には広範囲な研究資源の糾合が不可欠である。しかしながらフィールドに転がるサッカーボールの認識もままならない現実がある！。長い研究の歴史と厚い研究陣容を持つ視覚認識や認知科学の研究者の参入が極めて少ない事が原因であろう。これらの研究者が本格的に取り組めば急速にレベルアップが可能と思われる。参画が少ないのは、参画に対するモチベーションが不足している事が原因と考えられる。
RoboCupの名称は身体、頭脳ともに優れたロボットを目指す象徴的な名称であるが、同時にサッカーゲームの研究との印象を惹起する名称でもある。このため“あれは研究ではなく、遊びにすきない！”との評価を受ける危険を生じている。いわゆる正統派の参加が少なく、中には批判的に冷ややかな視線で眺めている人も多い。今一つはRoboCupは身体を伴う事で参画を困難にしている面がある。特にコンピュータ上での知識処理が中心のＡＩ研究者にとって、身体を作らないと参加出来ない、と言う事は参加への敷居の高いと思われる。しかしながらロボットの実現には知能がもっとも重要な要素である事を考えると大量のＡＩ研究者の参画は不可欠である。すなわちRoboCupの名称自体が光と影を作り出しているのである。前者の課題はサッカー以外のRoboCupレスキューをスタートさせ、後者はRoboCupシュミレーションで身体が無くても参画できる仕組みを作っている。しかしながらＡＩ研究との融合は解決が急がれる最も大きい課題であろう。

8 ヒューマノイドリーグの強化

人間共生ロボット実現の最も近道は、人間型ロボットを現実のものとしてその姿を示す事である。車輪型であったり、４足移動であったり、またおもちゃの様な超小型サイズのロボットでは、現実に我々の周囲で一緒に活動するとの実感を持つ事は困難である。直ちに人間サイズのヒューマノイドの実現には多大の困難が伴うが、できるだけ早い機会に人間サイズのロボットを実現する事が重要である。子供サイズのロボットからスタートするのが現実的である。人間のパートナーとしてのヒューマノイドが現実のものになろうとしていることを広く理解して頂く事が肝要である。単に歩き回るだけの能力であれば、その様なロボットを実現する事は現在の技術を前提にすればさほど困難ではない。有用な働きをするロボットに成長させるには、そのロボットに知恵、知識をどのように教え込むかと言う課題である。先に述べたように問題の中心は身体の実現ではなく、ロボットの知恵の実現である。標準化されて安価でロボットの身体や、或いは幾つかの自由度のもった集合関節と制御回路などを提供し、それをプラットホームとしてその上に感覚器官を含む認知機能や人工知能を搭載して【手足を持った知能】の実現が誰

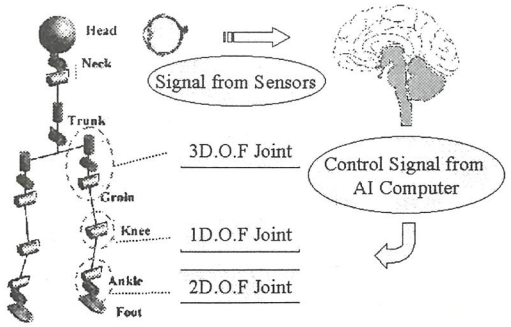


Fig-9 Standard Robot Body

にでも手軽にできる環境を整える事ができればロボト研究は飛躍的に加速される筈である。ロボット身体の製作の呪縛から開放すれば、ＡＩ研究者の参画障壁が無くなり、ロボットの知能の研究は飛躍的に進歩するものと思われる。

9 知識の獲得

コンピュータの急速な進歩により、人間と同等の能力をを持つロボットの実現も現実の問題として検討されている。しかしそれはあくまでも潜在能力であり、正しい認知能力や判断能力、及び各種運動能力を獲得して始めて人間と対等の能力を発揮する事が出来る。知恵を獲得していないロボットはいわゆる《木偶の坊》である。これは人間にとって有用などころか、反対に極めて危険な存在ですらある。教育や学習がロボットにとって何にも代え難い重要な事であるの所以である。ロボットが知恵を獲得するには Fig-10 に示すような手段がある。

Method	Contents
1.Teaching	Teaching by human or robot who have acquired knowledge already or knowledge server.
2.Learning	Acquire knowledge by self learning mimicry, trial/error, reinforced learning.
3.Instill and store	Inject knowledge via I/F from human or robot who have acquired knowledge already and store in the robot brain.
4.Instill momentary	Acquire knowledge via I/F from human or robot who have acquired knowledge already or knowledge server but not store in the robot brain.

Fig-10 Knowledge Acquisition by Robot

この中で、１，２の手段は人間が長年にわたり使用しており実績、経験も極めて豊富である。しかしながらこの方法は効率が悪く、長時間の忍耐も必要である。知識獲得の効率が悪いだけでなく、一人の人間が長年の努力で蓄積した知識も、その人の死と共に消滅してしまう重大な問題もあった。グーテンベルグの印刷術の発明で、文書による知識の流通は大幅に改善されたものの、伝達の効率が悪く、次に続く世代は殆んどゼロからやり直す必要があった。

一方ロボットは、新しく３，４の手段を使用する事で、従来より圧倒的に高い効率でロボットに知識を注入する事が出来るという、際立った特徴がある。知識をどの様な形式で蓄え、活用するかなどの知識処理、知識コンピュータの詳細な議論は、AI 研究の専門家の参画が不可欠である。

ロボットに貯えられた知識は時間、空間を越えて行き交い、ロボットの知識は急速に拡大し、人間のそれを越えるのも時間の問題となる。

さらに一つのロボットが保有する知識が、そのロボット自身の保有する記憶容量や演算能力によって制限されても、ネットワークを介し、ロボットの外にある知識サーバーのサポートを得る事が出来る。すなわちインターネットなど最近急速に発展している、グローバルなネットワークを活用すれば、全世界の知識さえも活用する事が可能で、実質的にロボットが活用できる知恵の総量は無限大である。このように知識が時間、空間を越えて流通できるため、古今東西の優れた知能を自在に活用できる事になる。人間が何万年もかけて獲得した知恵は、このようにロボットと言う新しい存在の誕生を契機として、さらに急速に拡大し『知の爆発！』が起るものと想像される。

10 RoboCup の向こうに見えるもの

RoboCup の挑戦的目標である『 2050 年までに人間のワールドカップ優勝チームに、勝つロボットチームを作る』が達成されると、そこに心身ともに優れた人間に似た【新しい存在】が誕生する事になる。高度な判断能力と抜群の身体能力を持つロボットは人間以上に我々の社会で活動する筈である。今迄人間が行っていた殆どの肉体作業を取って代わって行うようになり、それらの苦しい仕事から人間は完璧に開放される事になる。それだけでなく高度な知能を持つロボットは殆どの知識労働からも我々を開放してくれる筈である。この結果我々は真に創造的な仕事や趣味やスポーツなどの文化、芸術的な活動に多くの時間を割く事が出来るようになる。そこに至る過程でヒューマノイドの実現可能性と、我々のパートナーとしての有用性が広く認識されるようになる。その結果ますます多くの研究者、技術者の優秀な頭脳が投入され、ロボットの進歩・発展がますます加速される。更に実現への社会ニーズが高まるにつれ官民双方からの支援や産業界の取組みも開始され加速度的に開発と実用化が促進される。RoboCup プロジェクトの彼方には社会変革をも実現する理想社会の実現が見えている。

11 人類の幸福への貢献

現在に至るまで科学技術の発達は人類の幸福に大きく貢献してきた。翻ってロボット研究を眺めてみると工業用ロボットは生産性の向上と労力の低減に多大の貢献をしてきた。今後の活躍が期待されている【ユビキタスロボット】は社会のあらゆ

る場所へ進出し人間を助け、人類の幸福に大きく貢献する事が期待されている。アポロ計画より遥かに広い産業分野を巻き込み史上経験の無い巨大産業に成長する。さらに社会生活に革新的な影響を与え、人類の幸福への貢献は想像を絶する事にあると思われる。まさに現代の Super Big Challenge である。

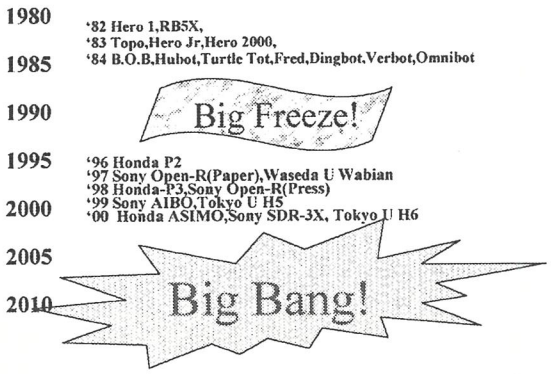


Fig-10 From Big-Freeze to Big-Bang

参考文献

[1] H. Kitano (Ed.) : RoboCup-97: Robot Soccer World Cup1, Lecture Note in Artificial Intelligence 1395, Springer (1998)
[2] M. Asada , H. Kitano (Eds.) : RoboCup-98: Robot Soccer World Cup2, Lecture Note in Artificial Intelligence 1604, Springer (1999)
[3] M. Veloso, E. Pagello, H. Kitano (Eds) : RoboCup-99: Robot Soccer World Cup3, Lecture Note in Artificial Intelligence 1856, Springer (2000)
[4] P. Stone, T. Balch, G. K. Kratzschmar (Eds) : RoboCup-00: Robot Soccer World Cup4, Lecture Note in Artificial Intelligence 2019, Springer (2001)
[5] H. Kitano, M. Asada: An Overview: RoboCup, today and tomorrow, IROS'99, Tutorial, TS4 (1999)
[6] M. Asada, S. Suzuki, M. Veloso, G. K. Kraetzschmar, and H. Kitano:What We Learned from RoboCup-97 and RoboCup-98, pp1426-1431, Proc. 1999 IEEE/RSJ Int'l. Conf. on Intelligent Robots and Systems (1999)
[7] H. Kitano, M. Asada:RoboCup Humanoid Chllenge:That's One Small Step for A Robot, One Giant Leap for Mankind, Proc. IROS-98 (1998)
[8] H. Kitano, M. Asada, I. Noda, H. Matsubara: RoboCup:Robot World Cup, pp30-36, IEEE Robotics & Automation Magazine (Sept. 1998)
[9] David A. Winter:Biomechanics and Motor Control of Human Movement, pp103-140, John Wiley & Sons (1990)
[10] P. M. Alexander:Exploring Biomechanics:Animals in Motion, W. H. Freeman (1992)
[11] H. Moravec: Robot, Oxford Univ. Press, pp91-126 (1999)
[12] R. Kurzweil: The Age of Intelligent Machines, MIT Press, pp425-458 (1992)
[13] R. Kurzweil: The Spiritual Machines, Penguin Books, pp101-132 (1999)
[14] H. Moravec: MIND Children, Harvard Univ. Press, pp51-74 (1988)
[15] M. Kosaka:Humanoids in Every Home, Proc. Humanoids2001, pp101-132 (2001)
[16] [ミニ特集] ヒューマノイド、日本ロボット学会誌、Vol. 15, No. 7 (1997)
[17] [特集] 人間共存型ロボット、日本ロボット学会誌、Vol. 16, No. 3 (1998)
[18] [特集] 人間共存・共存型ロボットシステム、日本ロボット学会誌、Vol. 19, No. 1 (2001)
[19] 吉川弘之 (監) : ロボットルネサンス、三田出版会 (1994)
[20] 吉川弘之、立花隆 : ロボットが街を歩く日、三田出版会 (1987)
[21] 白井良明、浅田稔 : 身近になるロボット、大阪大学出版会 (2001)
[22] 北野宏明 : 徹底ロボット学、ＰＨＰ出版 (2001)
[23] 北野宏明 : ﾛﾎﾞｶｯﾌﾟは日本版「ｱﾎﾟﾛ計画」だ、文芸春秋 6 月号、pp119-129 (1999)
[24] 瀬名秀明 : ロボット 21 世紀、文春新書、 (2001)
[25] 谷江和雄 : ロボットの効用に関する一考察、第 18 回 RSJ 講演会予稿集、pp905-906 (2000)
[26] 岡田美智男他 : 身体性とコンピュータ、共立出版、pp195-207 (2000)
[27] 小坂雅博 : 人間共生ロボットはなぜ 2 足歩行か、第 19 回 RSJ 講演会予稿集、pp905-906 (2001)
[28] 小坂雅博 : 人間共生ロボットはなぜ人間型か、第 19 回 RSJ 講演会予稿集、pp905-906 (2001)
[29] 小坂雅博 : 人間共生ロボットはなぜ万能型か、第 19 回 RSJ 講演会予稿集、pp905-906 (2001)
[30] 小坂雅博 : 人間共生ロボットの実現時期は、第 19 回 RSJ 講演会予稿集、pp905-906 (2001)
[31] 生命工学工業技術研究所、人体寸法データ集, 日本出版サービス (1996)
[21] 小原二郎他、人体を測る、日本出版サービス (1992)

Hidden Markov Modeling of Multi-Agent Systems and its Learning Method

NODA, Itsuki
Cyber Assist Reseach Center, National Institute of Advanced Industrial Science and Technology
PRESTO, Japan Science and Technology Corporation (JST)
I.Noda@aist.go.jp

Abstract

Two frameworks of hidden Markov modeling for multi-agent systems and its learning procedure are proposed. Although couple of variation of HMM have been proposed to model agents and their interactions, these models are not handle changes of environments, so that it is hard to handle behaviors of agents that act in dynamic environments like soccer. The proposed frameworks enables HMM to handles environment directly inside of state transitions. I first investigate a model that handle the environment in the same state transition of agent itself. In this model, the derived learning procedure can segment environment according to the tasks and behaviors the agent is performing. I also investigate a more structured model in which environment, agent, and another agent are treated as separated state transitions and coupled each other. For this model, in order to reduce the number of parameters, I introduce "symmetricity" among agents. Moreover, I discuss relation between reducing dependency in transitions and assumption of cooperative behaviors among multiple agents.

1 Introduction

When we train an agent or a team of agents (learner) to imitate behaviors of another agent or team (demonstrator), we must determine a framework to model agents or teams. Hidden Markov Model (HMM) is a popular candidate for this purpose. Because the behaviors of intelligent agents are complicated and structured, however, we should apply HMM carefully.

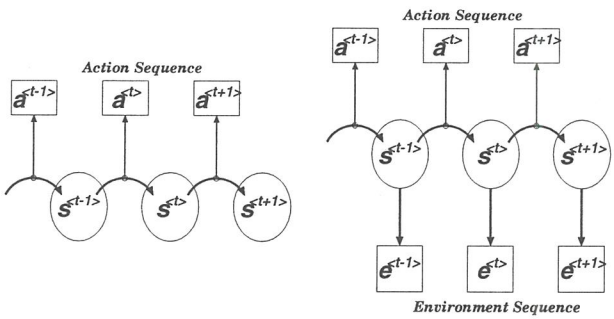
Suppose that we write a code of a reactive soccer agent by hands. We may write the following code for it:

```
while(true) {
  if (⟨is my role a passer ?⟩) {
    if (⟨is a receiver near ?⟩) {
      ⟨kick the ball to the receiver !⟩; ...
      ⟨change my role to receiver.⟩; }
    else if (⟨find dribbling course?⟩)...
  }
  else if (⟨is my role a receiver ?⟩) {
    if (⟨find an open space?⟩) {
      ⟨move to the space!⟩ }...  }
  else ...  }
```

As shown in this example, situations (environment and agent's internal states) are segmented into finite states like "*is a receiver near?*" and "*find dribbling course?*". These segmentations are vary according to agent's role or intention. In this example, "*is a receiver near?*" is not used when the agent's role is "*receiver*". In the context of the imitation learning, it is important to estimate what kind of segmentation of environment a demonstrator is using. The difficulty of the segmentation of environment is that the segmentation should change according to agent's current intention. This means that segmentation of environment should be acquired in the same time of learning intentions in the context of imitation learning.

In addition to it, when agents interact with each other, it is necessary to assume a kind of structure of states in HMM. In the above example, whole situations are classified into states according to roles of agents ("*passer*" and "*receiver*") and status of the environment ("*is a receiver near?*" and so on). Because it is difficult to acquire such structure through learning, it is better to use HMM in which states are structured suitably. In this case, we must pay attention the learning performance and reasonabilities of the structure.

In this article, I propose frameworks of HMM that can segment environment interaction between agents effectively through learning. In the following sections, I introduce a integrated HMM of agents and environment for learning of segmentation of environment in Section 2, and framework of HMM to represent interaction of agents in Section 3.



(a) Agent Model by Merly-type HMM.

(b) Agent Model by Moore-Merly-type HMM.

Figure 1: Agent HMM.

2 HMM for Agents in Dynamic Environment

2.1 Agent Model

In general, an autonomous agent is modeled as a Merly-type HMM (shown in Figure 1(a)), that is, the agent's behaviors are decided by the following manner:

- The agent has finite internal states.
- The internal state transites in a discrete time step.
- The next state ($s^{\langle t+1\rangle}$) is determined only by the previous state ($s^{\langle t\rangle}$).
- The agent' action ($a^{\langle t+1\rangle}$) is selected by the current state transition ($s^{\langle t\rangle}\rightarrow s^{\langle t+1\rangle}$).

This formalization lacks the effect of interaction between the agent and the environment. So, we introduce the following assumption:

- The internal state and the environment has a probabilistic relation.

This means that the environment ($e^{\langle t\rangle}$) can be determined by the internal state ($s^{\langle t\rangle}$) under the probabilistic relation ($Pr(e^{\langle t\rangle}|s^{\langle t\rangle})$). In other words, the changes of the environment can be handled as a Moore-type HMM (Figure 1(b)).

In summary, an agent and its environment can be defined as a following Moore-Merly-type HMM (MM-HMM).

$$\text{Agent} = \langle \boldsymbol{S}, \boldsymbol{A}, \boldsymbol{E}, \boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{R}, \pi \rangle ,$$

where $\boldsymbol{S} = \{s_i\}$ is a set of internal states, $\boldsymbol{A} = \{a_i\}$ is a set of action symbols, $\boldsymbol{E} = \{e_i\}$ is a set of environment symbols, $\boldsymbol{P} = \{p_{ij} = Pr(j^{\langle t+1\rangle}|i^{\langle t\rangle})|i,j \in \boldsymbol{S}, \forall t\}$ is a probability matrix of state transitions, $\boldsymbol{Q} = \{q_{ij}(a) = Pr(a^{\langle t+1\rangle}|i^{\langle t\rangle}, j^{\langle t+1\rangle})|i,j \in \boldsymbol{S}, a \in \boldsymbol{A}, \forall t\}$ is a probability tensor of actions for each transition, $\boldsymbol{R} = \{r_i(e) = Pr(e^{\langle t\rangle}|i^{\langle t\rangle})|i \in \boldsymbol{S}, e \in \boldsymbol{E}, \forall t\}$ is a probability vector of environment for each state, $\pi = \{\pi_i = Pr(i^{\langle 0\rangle})\}$ is a probability vector of initial states, and $\langle t\rangle$ on the right shoulder of a variable indicates time t.

2.2 Learning Algorithm

Suppose that a learner can observe a sequence of demonstrator's actions $\{a^{\langle 1\rangle} \ldots a^{\langle T\rangle}\}$ and changes of an environment $\{e^{\langle 0\rangle} \ldots e^{\langle T\rangle}\}$. The purpose of the learner is estimate an HMM that can explain the given action and environment sequences most likely.

For this HMM, the forward ($\alpha^{\langle t\rangle}(j)$) and backward ($beta^{\langle t\rangle}(i)$) probabilities are given by the following recursive formulas.

$$\alpha^{\langle t\rangle}(j) = \begin{cases} \pi_j r_j(e^{\langle 0\rangle}) & ; \quad t = 0 \\ \sum_{i\in\boldsymbol{S}} \alpha^{\langle t-1\rangle}(i) p_{ij} q_{ij}(a^{\langle t\rangle}) r_j(e^{\langle t\rangle}) & ; \quad otherwise \end{cases}$$

$$\beta^{\langle t\rangle}(i) = \begin{cases} 1 & ; \quad t = 0 \\ \sum_{j\in\boldsymbol{S}} p_{ij} q_{ij}(a^{\langle t+1\rangle}) r_j(e^{\langle t+1\rangle}) \beta^{\langle t+1\rangle}(j) & ; \quad otherwise \end{cases}$$

Using these probabilities, extended Baum-Welch algorithm to adjust p_{ij}, $q_{ij}(a)$, $r_i(e)$ and π_i are derived as follows:

$$p_{ij} \leftarrow \frac{\sum_t \xi^{\langle t\rangle}(i,j)}{\sum_t \gamma^{\langle t-1\rangle}(i)} \qquad q_{ij}(a) \leftarrow \frac{\sum_{t|a^{\langle t\rangle}=a} \xi^{\langle t\rangle}(i,j)}{\sum_t \xi^{\langle t-1\rangle}(i,j)}$$

$$r_i(e) \leftarrow \frac{\sum_{t|e^{\langle t\rangle}=e} \gamma^{\langle t\rangle}(i)}{\sum_t \gamma^{\langle t\rangle}(i)} \qquad \pi_i \leftarrow \gamma^{\langle 0\rangle}(i)$$

where

$$\xi^{\langle t\rangle}(i,j) = \frac{\alpha^{\langle t-1\rangle}(i) p_{ij} q_{ij}(a^{\langle t\rangle}) r_j(e^{\langle t\rangle}) \beta^{\langle t\rangle}(j)}{P(\boldsymbol{a}^{\langle *\rangle}, \boldsymbol{e}^{\langle *\rangle}|\text{Agent})}$$

$$\gamma^{\langle t\rangle}(j) = \frac{\alpha^{\langle t\rangle}(j) \beta^{\langle t\rangle}(j)}{P(\boldsymbol{a}^{\langle *\rangle}, \boldsymbol{e}^{\langle *\rangle}|\text{Agent})}$$

2.3 Segmentation of Environments

When the above learning succeeds, the learner gets a suitable HMM, whose state transition reflects both of the internal intentions of the demonstrator and segmentation of the environment. In the HMM, each state corresponds a combination of an intention and a segmentation for it. In other words, the representation of the intention and the segmentation are mixed in a set of states. While such representation is enough to imitate demonstrator's behavior, it will be still useful to know how the acquired HMM segments the environment.

The segmentation of the environment can be represented by a probability function $Pr(s|e)$ where e is an environment data and s is an internal state of HMM. This probability can be calculated by the following equation:

$$Pr(s|e) = \frac{Pr(e|s)Pr(s)}{Pr(e)} = \frac{r_s(e)Pr(s)}{Pr(e)}$$

Using this equation, we can know how the acquired HMM segments the environment.

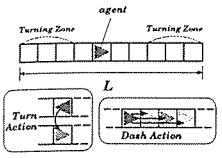
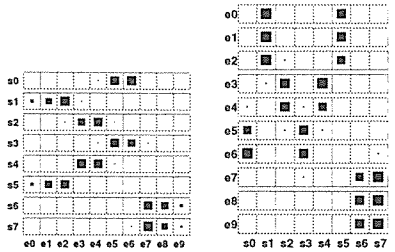


Figure 2: Setting of Ex.1: Discrete World



(a) $Pr(e|s)$ (b) $Pr(s|e)$

Figure 3: Result of Ex.1: Discrete World (L=10)

2.4 Experimental Result

In order to show the performance of the proposed method to acquire separation of environments, I conducted the following experiments.

2.4.1 Ex.1: Discrete World

In the first experiment, we use a linear block world shown in Figure 2. An agent is moving in the world using *dash* and *turn* actions. When the agent *turn*s, the agent's direction flips between left and right. When the agent *dash*es, the agent's position moves forward according to its direction. The step size of a dash action varies 1 to 3 randomly.

The task of a learner is to acquire the rule of another agent (demonstrator) who behaves as described below. A learner can observe the demonstrator's position (= environment, $\{e^{\langle t \rangle}\}$) and action ($\{a^{\langle t \rangle}\}$) in each time t. We suppose that the demonstrator behaves according to the following rules:

- If the position is in the left/right turning zone (margin of the zone is 3) and the direction is left/right, then turn.
- Otherwise, dash.

The main point of this experiment is that whether the learner can acquire the correct segmentation by which the turning zone will be represented explicitly in the state transition, because the concept of the turning zone is unobservable to the learner. Also, estimation of the dash step size is also important, because it defines how the world (environment) should be segmented in order to simulate it by an HMM. Note that demonstrator's direction is not observable. This means that the learner needs to acquire the representation of the direction through the learning.

In addition to it, the learning of general HMMs is not guaranteed to reach the global optimal solution: the adaptation may fall down to a local optimum. Therefore, we executed the above learning procedure using different initial parameters 100 times, calculated the average of the likelihood of given example sequences, and select the best one as the result.

Figure 3 shows the result of the experiment. In this figure, (a) shows possibilities of environment symbols ($e_0 \ldots e_9$) for each state ($s_0 \ldots s_7$). Each box corresponds to $Pr(e_i|s_j)$ whose value is denoted the size of black area in the box. For example, in the s_0 line of boxes, columns of e_5 and e_6 have significant values and both values are relatively equal. This can be interpreted that e_5 and e_6 are grouped in the same state s_0, because the environment is estimated e_5 or e_6 equally when the HMM is in state s_0.

Similarly, Figure 3-(b) shows possibilities of states for each environment symbol, whose value $Pr(s_j|e_i)$ is denoted black area of each box. For example, the e_0 line has two columns of s_1 and s_5 who have relatively the same significant possibilities. This can be interpreted that e_0 can correspond to two states, s_1 and s_5, equally.

We can extract the following points from this result:

- The acquired HMM segments the environment into 4 regions, $\{e_0, e_1, e_2\}$, $\{e_7, e_8, e_9\}$, $\{e_3, e_4\}$, and $\{e_5, e_6\}$. The first two regions correspond the "turning zone" defined inside of the demonstrator. Rest of the two regions have the same length, 2. This value correspond the average step size of dash commands. These results means that the acquired HMM represents both of the rules of agent behavior and dynamics of the environment.
- Each environment symbol corresponds two states. This means that the HMM recognizes the (unobservable) direction of the demonstrator by doubled states for each environment.

2.4.2 Ex.2: Continuous World

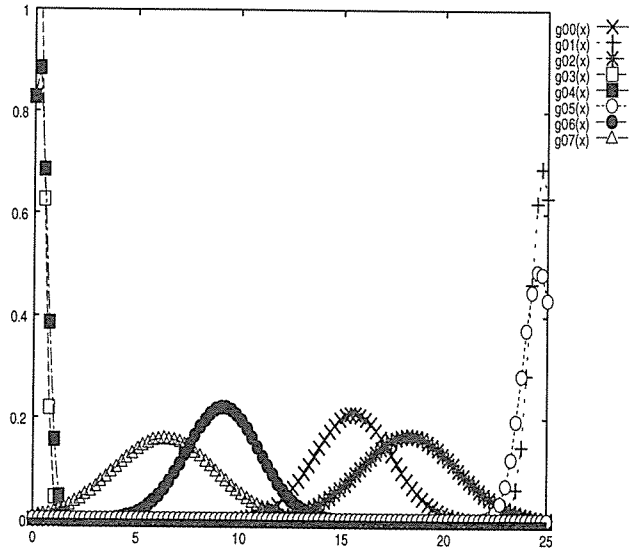
The second experiment is a continuous version of the previous experiment. In this experiment, the demonstrator's position (e) is a continuous value instead of a discrete symbol. The rule of the demonstrator's behavior is the same as the previous one. The length of the world L is 25.0 and step size of a dash varies 5.0 to 15.0 randomly.

Figure 4 shows the acquired $Pr(e|s)$ and $Pr(s|e)$ after the learning. In these graphs, the probabilities are plotted as follows:

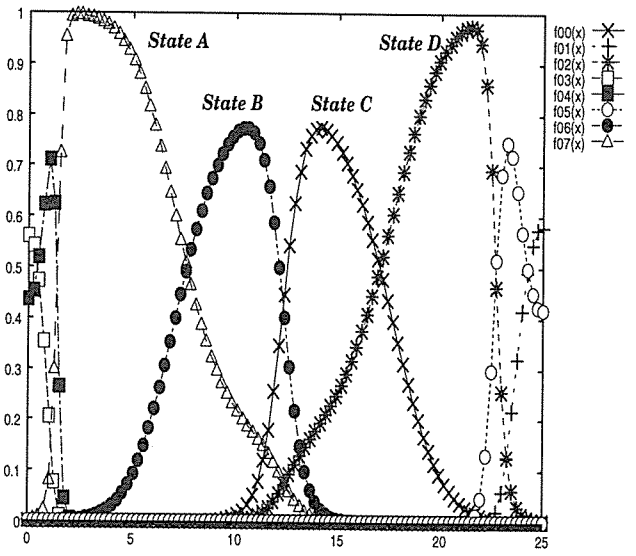
- $Pr(e|s)$ is plotted as a set of probability density functions of environment value, $g_s(e) = Pr(e|s)$, for each state s.
- $Pr(s|e)$ is plotted as a set of changes of probability of each state, $f_s(e) = Pr(s|e) = Pr(e|s)Pr(s)/Pr(e)$, according to environment value.

In the figure, (a.1) and (a.2) show the result when the number of HMM's states is 8, and (b) shows the result in the case of 14. From (a.1) and (a.2), we can find that the HMM segments the environment in the similar way as the discrete case of the previous experiment as follows:

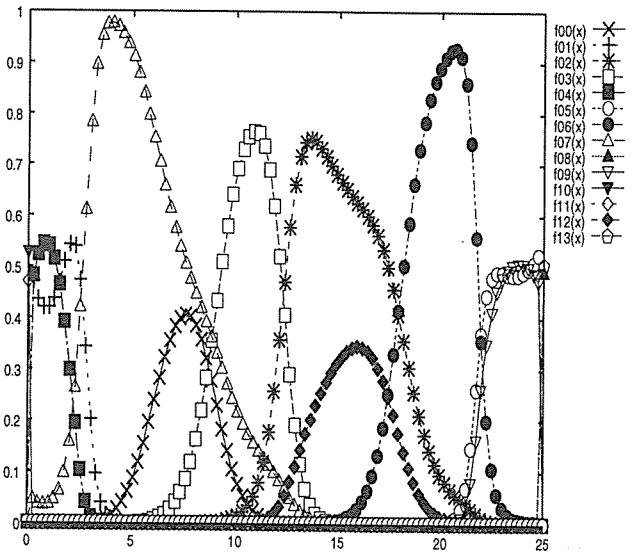
- Two turning zones at the both ends of the world are segmented clearly.
- There are two corresponding states for the most of the environment value. This means that the HMM represents the direction of movement by the two states.



(a.1) $Pr(e|s)$ (# of states = 8)



(a.2) $Pr(s|e)$ (# of states = 8)



(b) $Pr(s|e)$ (# of states = 14)

Figure 4: Result of Ex.2:

We also can find an additional features from these graphs: There are 4 peaks (State A–D in (a.2)) in the middle of the environment in the graph, and, A and B (or C and D) are relatively overlapped with each other. This means that the two states A and B (or C and D) indicate difference of the direction of the movement as mentioned above. While two states are overlapped completely in the first experiment, however, peaks of A and B (or C and D) are shifted. As a result, the segment point of the environment in each direction are different. For example, the segment point [1] is about 13 in the case that the direction is right (the case transition is B $\rightarrow$D), and it is about 11 in the case of left (C $\rightarrow$A) [2] This means that the segmentation of the environment is not fixed for all agent's internal states, but varies depend on them. The structure of the segmentation are stable when we use more states in the learning. For example, when we use 14 states to learn the same task, the HMM can acquire the similar segmentation of the environment (Figure 4-(b)).

In order to show that the proposed method can acquire segmentation of environment flexibly, we conducted the following experiment. We introduce an additional *half turning zone* in the middle of the world, in which the demonstrator turns in the probability 0.5 . The detailed rule in this turning zone is as follows:

- If the demonstrator's direction is right and the position is in the left hand side of the *half turning zone*, then turn in the probability 0.5.
- If the direction is left and the position is in the right hand side of the zone, then turn in the probability 0.5.

Figure 5 shows how the segmentation of the environment ($Pr(s|e)$) changes by using the various numbers of states. As shown in this result, many states are used to represent turning zones, especially the half turning zone (the center area of the world). We can see when the number of state increase, the HMM assign many states to segment the turning zones, especially the half turning zone. This is because that the conditions to decide demonstrator's behaviors are complex and use detailed information about the environment.

2.5 Discussion: Environment as Output

The proposed method looks little bit strange because it handles environment as output from states rather than as input to state transitions. Input-output HMMs seems more reasonable to model relations between agents and environments, in which the environment is treated as input to state-transitions [Bengio and Frasconi, 1995, Jordan *et al.*, 1997a]. There are the following different points between these two methods:

- When we handle the environment as input, we can apply the HMM for planning. Suppose that initial and goal situations of environment ($e^{\langle 0 \rangle}$ and $e^{\langle T \rangle}$)

[1] a crossing point of probabilities of two states.

[2] The transition B $\rightarrow$D and C $\rightarrow$A are extracted from probability matrix of state transition of the trained HMM.

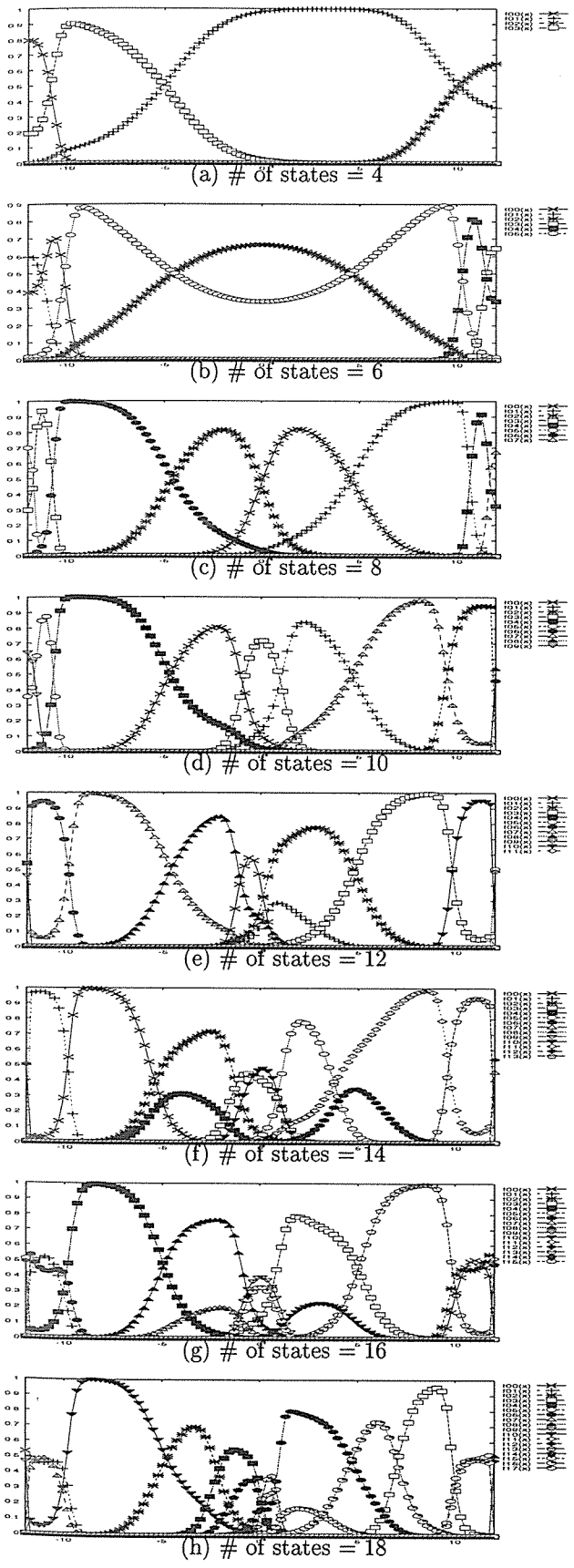
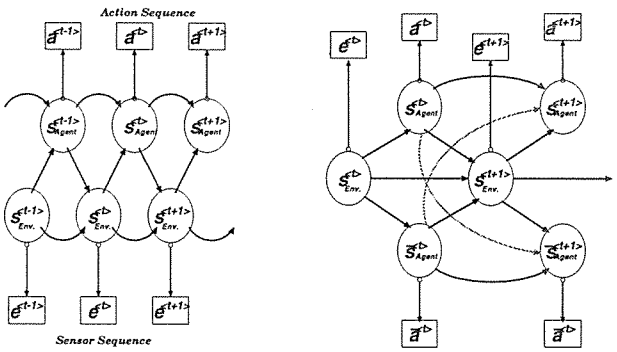


Figure 5: Changes of Segmentation ($Pr(s|e)$) by the Number of States (in Ex.2)



(a) Agent and Environment (b) Two Agents and Environment

Figure 6: Coupled HMMs of Agents and Environment.

are given. Then, the planning can be formalized as follows:

> To get the most likely path of state transitions that maximize the probability $Pr(e^{\langle 0\rangle}, e^{\langle T\rangle}|\text{Agent})$.

When the environment is handled as output like the proposed method, we can seek the most likely path simply using well-known algorithm like Viterbi's one. On the other hand, we need an additional simulator or inverse model of environment when the environment is handled as input.

- When we use continuous value for input of HMM, we need to use gradient ascent to learn the parameters in a cycle, which requires more computation power. On the other hand, in the proposed method, we can apply the one-shot adaptation algorithm derived in Section 2.2.

3 Symmetrically Coupled HMM

3.1 Symmetricity Assumption

In Section 2, we handle environment and agent's intention by a single HMM. However, the number of states increases exponentially when the agent has more complex intentions. This is significant when HMM handles interactions among agents. In this case, we will face *generalization performance problem.* As the number of states or learning parameters increase, the huge number of examples are required to guarantee the generalization performance. In order to avoid this problem, I introduce *symmetricity assumption* among agents as follows:

> **symmetricity assumption**
> Every agent has the same rules of behavior. In other words, every agent shares the same state transition rules with each other.

To reflect the above assumption in HMM, first, I divide the internal state into two states, environment state $s_{\mathbf{e}}$ and agent state $s_{\mathbf{a}}$, and form a coupled HMM as shown in Figure 6-(a). In this model, sensor data $e^{\langle t\rangle}$ and action commands $a^{\langle t\rangle}$ are determined by environment states $s_{\mathbf{e}}^{\langle t\rangle}$

and agent states $s_{\mathbf{a}}^{\langle t \rangle}$ respectively. Transitions of both states are determined as follows: The next environment state $s_{\mathbf{e}}^{\langle t+1 \rangle}$ is determined according to the current environment and agent states $\{s_{\mathbf{e}}^{\langle t \rangle}, s_{\mathbf{a}}^{\langle t \rangle}\}$. The next agent state $s_{\mathbf{a}}^{\langle t+1 \rangle}$ is determined according to the current agent state and the new environment $\{s_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{e}}^{\langle t+1 \rangle}\}$. Then I introduce the second agent who cooperates with the first agent as shown in Figure 6-(b). In this coupling, both state transitions becomes affected by the second agent state $\bar{s}_{\mathbf{a}}^{\langle t \rangle}$. This is summarized as the probabilities of state transitions as follows:

$$\begin{aligned} Pr(s_{\mathbf{e}}^{\langle t+1 \rangle}|*) &= Pr(s_{\mathbf{e}}^{\langle t+1 \rangle}|s_{\mathbf{e}}^{\langle t \rangle}, s_{\mathbf{a}}^{\langle t \rangle}, \bar{s}_{\mathbf{a}}^{\langle t \rangle}) \\ Pr(s_{\mathbf{a}}^{\langle t+1 \rangle}|*) &= Pr(s_{\mathbf{a}}^{\langle t+1 \rangle}|s_{\mathbf{a}}^{\langle t \rangle}, \bar{s}_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{e}}^{\langle t+1 \rangle}) \end{aligned}$$

In order to complete the state transition for Figure 6-(b), we must consider about transitions of the second agent state $\bar{s}_{\mathbf{a}}$. Here, I apply *symmetricity assumption* for the second state transition, that is, the probabilities of state transitions of the second agent are determined by the same one of the first agent state transitions. The most naive implementation of this assumption is that the probabilities are described as follows:

$$\begin{aligned} Pr(\bar{s}_{\mathbf{a}}^{\langle t+1 \rangle}|*) &= Pr(\bar{s}_{\mathbf{a}}^{\langle t+1 \rangle}|\bar{s}_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{e}}^{\langle t+1 \rangle}) \\ &= Pr(s_{\mathbf{a}}^{\langle t+1 \rangle} = \bar{s}_{\mathbf{a}}^{\langle t+1 \rangle}| \\ &\qquad s_{\mathbf{a}}^{\langle t \rangle} = \bar{s}_{\mathbf{a}}^{\langle t \rangle}, \bar{s}_{\mathbf{a}}^{\langle t \rangle} = s_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{e}}^{\langle t+1 \rangle}) \end{aligned}$$

This formulation is valid when both agents share the same environment state. In general, however, two agents may have different environment state inside of them, because the environment state in this formalization is a kind of internal world state that each agent has. Such a situation is not avoidable especially when the sensor data $e^{\langle t \rangle}$ is represented from the viewpoint of each agent. In order to overcome this problem, I propose a *symmetrically coupled HMM* (sCHMM) shown in Figure 7. In this model, the second agent has its own environment state $\bar{s}^{\langle t \rangle}$. Using this, the transition of $\bar{s}_{\mathbf{a}}^{\langle t \rangle}$ are represented as follows:

$$\begin{aligned} Pr(\bar{s}_{\mathbf{a}}^{\langle t+1 \rangle}|*) &= Pr(s_{\mathbf{a}}^{\langle t+1 \rangle} = \bar{s}_{\mathbf{a}}^{\langle t+1 \rangle}| \\ &\qquad s_{\mathbf{a}}^{\langle t \rangle} = \bar{s}_{\mathbf{a}}^{\langle t \rangle}, \bar{s}_{\mathbf{a}}^{\langle t \rangle} = s_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{e}}^{\langle t+1 \rangle} = \bar{s}_{\mathbf{e}}^{\langle t+1 \rangle}), \end{aligned}$$

where the transition of the second environment state $\bar{s}^{\langle t \rangle}$ follows:

$$\begin{aligned} Pr(\bar{s}_{\mathbf{e}}^{\langle t+1 \rangle}|*) &= Pr(s_{\mathbf{e}}^{\langle t+1 \rangle} = \bar{s}_{\mathbf{e}}^{\langle t+1 \rangle}| \\ &\qquad s_{\mathbf{e}}^{\langle t \rangle} = \bar{s}_{\mathbf{e}}^{\langle t \rangle}, s_{\mathbf{a}}^{\langle t \rangle} = \bar{s}_{\mathbf{a}}^{\langle t \rangle}, \bar{s}_{\mathbf{a}}^{\langle t \rangle} = s_{\mathbf{a}}^{\langle t \rangle}) \end{aligned}$$

3.2 Formalization and Learning Procedure

I summarize the sCHMM agent as the following tuple:

$$\text{Agent} = \langle \boldsymbol{S}_{\mathbf{e}}, \boldsymbol{S}_{\mathbf{a}}, \boldsymbol{E}, \boldsymbol{A}, \boldsymbol{P}_{\mathbf{e}}, \boldsymbol{P}_{\mathbf{a}}, \boldsymbol{Q}_{\mathbf{e}}, \boldsymbol{Q}_{\mathbf{a}}, \pi_{\mathbf{e}}, \pi_{\mathbf{a}} \rangle ,$$

where $\boldsymbol{S}_{\mathbf{a}} = \{s_{\mathbf{ai}}\}$ and $\boldsymbol{S}_{\mathbf{e}} = \{s_{\mathbf{ei}}\}$ are sets of states for agent and environment respectively, $\boldsymbol{E} = \{e_i\}$ is a set of sensed environment symbols, and $\boldsymbol{A} = \{a_i\}$ is a set of agent action symbols. $\boldsymbol{P}_{\mathbf{e}} = \{p_{\mathbf{eijkl}}|i \in \boldsymbol{S}_{\mathbf{e}}v, j, k \in \boldsymbol{S}_{\mathbf{a}}g, \forall t\}$ and $\boldsymbol{P}_{\mathbf{a}} = \{p_{\mathbf{ajklm}}|j, k \in \boldsymbol{S}_{\mathbf{a}}g, m \in \boldsymbol{S}_{\mathbf{e}}v, \forall t\}$ are

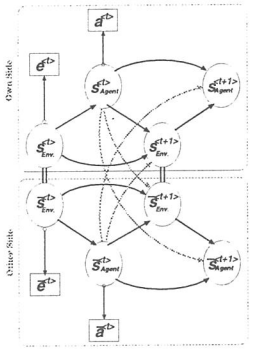


Figure 7: Symmetrically Coupled HMM.

probability tensors of state transitions of environment and agent, $\boldsymbol{Q}_{\mathbf{e}} = \{q_{\mathbf{ei}}(e)|i \in \boldsymbol{S}_{\mathbf{e}}, e \in \boldsymbol{E}, \forall t\}$ and $\boldsymbol{Q}_{\mathbf{a}} = \{q_{\mathbf{aj}}(a)|j \in \boldsymbol{S}_{\mathbf{a}}, a \in \boldsymbol{A}, \forall t\}$ are probability tensors of is observed symbols of environment and actions, and $\boldsymbol{\pi}_{\mathbf{e}} = \{\pi_{\mathbf{ei}} = Pr(s_{\mathbf{e}}^{\langle 0 \rangle} = i)|i \in \boldsymbol{S}_{\mathbf{e}}\}$ and $\boldsymbol{\pi}_{\mathbf{a}} = \{\pi_{\mathbf{aj}} = Pr(s_{\mathbf{a}}^{\langle 0 \rangle} = j)|j \in \boldsymbol{S}_{\mathbf{a}}\}$ are probability vectors of initial states of environment and agent. Each element of $\boldsymbol{P}_{\mathbf{e}}$, $\boldsymbol{P}_{\mathbf{a}}$, $\boldsymbol{Q}_{\mathbf{e}}$, and $\boldsymbol{Q}_{\mathbf{a}}$ represents the following probability.

$$\begin{aligned} p_{\mathbf{eijkl}} &= Pr(s_{\mathbf{e}}^{\langle t \rangle} = l \mid s_{\mathbf{e}}^{\langle t-1 \rangle} = i, s_{\mathbf{a}}s^{\langle t-1 \rangle} = j, s_{\mathbf{a}}o^{\langle t-1 \rangle} = k \\ p_{\mathbf{ajklm}} &= Pr(s_{\mathbf{a}}^{\langle t \rangle} = m \mid s_{\mathbf{a}}s^{\langle t-1 \rangle} = j, s_{\mathbf{a}}o^{\langle t-1 \rangle} = k, s_{\mathbf{e}}^{\langle t \rangle} = l \\ q_{\mathbf{ei}}(e) &= Pr(e^{\langle t \rangle} = e \mid s_{\mathbf{e}}^{\langle t \rangle} = i) \\ q_{\mathbf{aj}}(a) &= Pr(a^{\langle t \rangle} = a \mid s_{\mathbf{a}}^{\langle t \rangle} = i) \end{aligned}$$

We can derive a learning procedure for sCHMM as shown below. Suppose that sequences of sensor information $\{e^{\langle t \rangle}\}$, agent's own actions $\{a^{\langle t \rangle}\}$, and other's actions $\{\bar{a}^{\langle t \rangle}\}$ are observed $(0 \leq t < T)$. We can calculate agent's own forward and backward probabilities, $\alpha_{lmn}^{\langle t \rangle}$ and $\beta_{ijk}^{\langle t \rangle}$ respectively, as follows:

$$\alpha_{lmn}^{\langle t \rangle} = \begin{cases} \pi_{\mathbf{el}}\pi_{\mathbf{am}}\pi_{\mathbf{an}}Q_{(lmn)}(W^{\langle 0 \rangle}) \\ \qquad ; \quad t = 0 \\ \sum_{(ijk)} \alpha_{ijk}^{\langle t-1 \rangle} P_{(ijk)(lmn)} Q_{(lmn)}(W^{\langle t \rangle}) \\ \qquad ; \quad otherwise \end{cases}$$

$$\beta_{ijk}^{\langle t \rangle} = \begin{cases} 1 \\ \qquad ; \quad t = T-1 \\ \sum_{(lmn)} P_{(ijk)(lmn)} Q_{(lmn)}(W^{\langle t+1 \rangle}) \beta_{lmn}^{\langle t+1 \rangle} \\ \qquad ; \quad otherwise \end{cases},$$

where

$$\begin{aligned} P_{(ijk)(lmn)} &= p_{\mathbf{eijkl}} \cdot p_{\mathbf{ajklm}} \cdot p_{\mathbf{akjln}} \\ Q_{(ijk)}(W^{\langle t \rangle}) &= Q_{(ijk)}(e^{\langle t \rangle}, a^{\langle t \rangle}, \bar{a}^{\langle t \rangle}) \\ &= q_{\mathbf{ei}}(e^{\langle t \rangle}) \cdot q_{\mathbf{aj}}(a^{\langle t \rangle}) \cdot q_{\mathbf{ak}}(\bar{a}^{\langle t \rangle}) \end{aligned}$$

In the same way, other's forward and backward prob-

abilities, $\bar{\alpha}^{\langle t \rangle}_{lmn}$ and $\bar{\beta}^{\langle t \rangle}_{ijk}$ respectively, can be calculated:

$$\bar{\alpha}^{\langle t \rangle}_{lnm} = \begin{cases} \pi_{\mathbf{el}}\pi_{\mathbf{am}}\pi_{\mathbf{an}}Q_{(lnm)}(\bar{W}^{\langle 0 \rangle}) \\ \quad ; \quad t = 0 \\ \sum_{(ikj)} \bar{\alpha}^{\langle t-1 \rangle}_{ikj} P_{(ikj)(lnm)} Q_{(lnm)}(\bar{W}^{\langle t \rangle}) \\ \quad ; \quad otherwise \end{cases}$$

$$\bar{\beta}^{\langle t \rangle}_{ikj} = \begin{cases} 1 \\ \quad ; \quad t = T-1 \\ \sum_{(lnm)} P_{(ikj)(lnm)} Q_{(lnm)}(\bar{W}^{\langle t+1 \rangle}) \beta^{\langle t+1 \rangle}_{lnm} \\ \quad ; \quad otherwise \end{cases},$$

where $\bar{W}^{\langle = \rangle}\{\bar{e}^{\langle t \rangle}, \bar{a}^{\langle t \rangle}, a^{\langle t \rangle}\}$ and $\bar{e}^{\langle t \rangle}$ is the sensor information received by the second agent. Using these probabilities, we can adapt transition and output probabilities $p_{\mathbf{eijkl}}, p_{\mathbf{ajklm}}, q_{\mathbf{ei}}, q_{\mathbf{ej}}$ as follows:

$$\begin{aligned} p_{\mathbf{eijkl}} &\leftarrow \sum_m \sum_n \hat{P}_{(ijk)(lmn)} \\ p_{\mathbf{ajklm}} &\leftarrow \frac{\sum_i \sum_n \hat{P}_{(ijk)(lmn)}}{\sum_i p_{\mathbf{eijkl}}} \\ q_{\mathbf{ei}}(e) &\leftarrow \sum_j \sum_k \sum_a \sum_{\bar{a}} \hat{Q}_{(ijk)}(e, a, \bar{a}) \\ q_{\mathbf{aj}}(a) &\leftarrow \sum_i \sum_k \sum_e \sum_{\bar{a}} \hat{Q}_{(ijk)}(e, a, \bar{a}), \end{aligned}$$

where

$$\hat{P}_{(ijk)(lmn)} = \frac{\sum_t \xi^{\langle t \rangle}_{(ijk)(lmn)} + \sum_t \bar{\xi}^{\langle t \rangle}_{(ikj)(lnm)}}{\sum_t \gamma^{\langle t-1 \rangle}_{ijk} + \sum_t \bar{\gamma}^{\langle t-1 \rangle}_{ikj}} \quad (1)$$

$$\hat{Q}_{(ijk)}(W) = \frac{\sum_{t, W^{\langle t \rangle} = W} \gamma(ijk)^{\langle t \rangle} + \sum_{t, W^{\langle t \rangle} = W} \bar{\gamma}^{\langle t \rangle}_{ikj}}{\sum_t \gamma(ijk)^{\langle t \rangle} + \sum_t \bar{\gamma}^{\langle t \rangle}_{ikj}} \quad (2)$$

$$\begin{aligned} \xi^{\langle t \rangle}_{(ijk)(lmn)} &= \alpha^{\langle t-1 \rangle}_{(ijk)} P_{(ijk)(lmn)} Q_{(lmn)}(W^{\langle t \rangle}) \beta^{\langle t \rangle}_{(lmn)} \\ \bar{\xi}^{\langle t \rangle}_{(ikj)(lnm)} &= \bar{\alpha}^{\langle t-1 \rangle}_{(ikj)} P_{(ikj)(lnm)} Q_{(lnm)}(\bar{W}^{\langle t \rangle}) \bar{\beta}^{\langle t \rangle}_{(lnm)} \\ \gamma^{\langle t \rangle}_{(lmn)} &= \alpha^{\langle t \rangle}_{(lmn)} \beta^{\langle t \rangle}_{(lmn)} \\ \bar{\gamma}^{\langle t \rangle}_{(lnm)} &= \bar{\alpha}^{\langle t \rangle}_{(lnm)} \bar{\beta}^{\langle t \rangle}_{(lnm)} \end{aligned}$$

3.3 Discussion: The Number of Parameters in the Model

As mentioned before, the number of parameters in HMM is an important factor for generalization performance of learning. In the case of coupled HMM, especially, the number of parameters increase exponentially. Actually, if we use the model shown in Figure 6-(b) without symmetricity assumption, the number of parameters in the state transition is

$$|S_{\mathbf{e}}|^2 |S_{\mathbf{a}}|^N + N |S_{\mathbf{e}}| |S_{\mathbf{a}}|^{N+1}$$

where N is the number of agents. This is already reduced from the number of parameters $(|S_{\mathbf{e}}| |S_{\mathbf{a}}|^N)^2$ in the case we represent the same model using single HMM. Compared with this, symmetrically coupled HMM has fewer parameters as follows:

$$|S_{\mathbf{e}}|^2 |S_{\mathbf{a}}|^N + |S_{\mathbf{e}}| |S_{\mathbf{a}}|^{N+1}$$

In addition to it, the symmetricity assumption increase the virtual number of examples. Eq. 1 and Eq. 2 mean that the same HMM is trained by using both pairs of $\{e^{\langle t \rangle}, a^{\langle t \rangle}\}$ and $\{\bar{e}^{\langle t \rangle}, \bar{a}^{\langle t \rangle}\}$ for a given observation $\{e^{\langle t \rangle}, a^{\langle t \rangle}, \bar{a}^{\langle t \rangle}\}$. As a result, the generalization performance of learning is improved by the virtually doubled examples.

It is, however, true that an sCHMM still has too many parameters for real applications. Therefore, it is meaningful to introduce additional assumptions to reduce the number of parameter. Fortunately, in the case of cooperative interaction in the multi-agent systems, we can pick-up reasonable assumptions as follows:

- "*no explicit communication*" assumption: In the formalization of sCHMM, the transition of the agent state is affected by the previous states of other agents. This means that agents use explicit communication with each other. In the case of human cooperative behaviors like real soccer, on the other hand, we do not use so much explicit communication, but model others via sensor information instead. In such case, the transition of the agent state can be represented as follows:

  $$Pr(s^{\langle t+1 \rangle}_{\mathbf{a}}|*) = Pr(s^{\langle t+1 \rangle}_{\mathbf{a}}|s^{\langle t \rangle}_{\mathbf{a}}, s^{\langle t+1 \rangle}_{\mathbf{e}})$$

  In this case, the total number of the parameters is reduced to:

  $$|S_{\mathbf{e}}|^2 |S_{\mathbf{a}}|^N + |S_{\mathbf{e}}| |S_{\mathbf{a}}|^2$$

- "*filtering*" assumption: Usually, when we write a code of agent behavior, we classify states systematically. For example, in the code shown in Section 1 states are grouped by agent's roles (agent states) first then branched by world status (environment states) second. This can be represented by the following manner in the transition of HMM:

  $$Pr(s^{\langle t+1 \rangle}_{\mathbf{a}}|*) = Pr(s^{\langle t+1 \rangle}_{\mathbf{a}}|s^{\langle t+1 \rangle}_{\mathbf{e}}) \cdot Pr(s^{\langle t+1 \rangle}_{\mathbf{a}}|s^{\langle t \rangle}_{\mathbf{a}})$$

  In this case, the number of parameters are reduced to:

  $$|S_{\mathbf{e}}|^2 |S_{\mathbf{a}}|^N + |S_{\mathbf{e}}| |S_{\mathbf{a}}| + |S_{\mathbf{a}}|^{N+1}$$

- "*shared joint intention*" assumption: During a cooperation of multiple agents each agent believes that all agents share the joint intention. This means that each agent believes that other agents will behave as it wants. In this case, the transition of environment states can be represented as follows:

  $$Pr(s^{\langle t+1 \rangle}_{\mathbf{e}}|*) = Pr(s^{\langle t+1 \rangle}_{\mathbf{e}}|s^{\langle t \rangle}_{\mathbf{e}}, s^{\langle t \rangle}_{\mathbf{a}})$$

  This will reduce the number of parameters to:

  $$|S_{\mathbf{e}}|^2 |S_{\mathbf{a}}| + |S_{\mathbf{e}}| |S_{\mathbf{a}}|^{N+1}$$

Note that this assumption can not be applied with the "*no explicit communication*" assumption, because the sCHMM is reduced into a simple CHMM like Figure 6-(a) that does not reflect cooperation among agents.

3.4 Related Works

Brand Et al. [Brand, 1997, hidden Markov models for complex action recognition, 1996] proposed coupled HMM and its learning method, in which several HMMs are coupled via inter-HMM dependencies. Jordan Et al. [Jordan *et al.*, 1997b, Ghahramani and Jordan, 1997, Jordan *et al.*, 1999] proposed factorial HMM and hidden Markov decision trees. Both of works mainly focused on reducing the complexity in EM processes. Even using these HMMs, the complexity of calculation of a naive implementation increase exponentially, so that it is hard to handle the large number of states. They use mean field approximation or N-heads dynamic programming to reduce the cost of the approximation of posterior probabilities. However, they does not focused on symmetricity in agent-interactions and generalization performance problem.

These methods can be applicable to our model. Actually, a naive implementation of learning method derived in the previous section costs $O(TN^4M^2)$, which is too huge for dynamical application like soccer. Above methods will reduce the cost into $O(TN^2M)$, which is reasonable cost for real application.

4 Concluding Remarks

In this article, we proposed two frameworks to learn behaviors of multi-agents in dynamic environment using HMM. The first framework handles agent's environments as output of HMM rather than as input. As the result, the acquired HMM represents segmentation of environment explicitly in the states. The explicit segmentation leads the following features to the HMM:

- HMM can be used planning of agent's behavior working in a dynamic environment.
- Flexible segmentation can improve generalization performance of the learning.

The second framework is conducted to represent interaction of multi-agents and environments. In order to avoid the explosion of the number of parameters, I introduced symmetricity assumptions among agents, and propose symmetrically coupled HMM (sCHMM) and its learning procedure.

There are the following open issues on the proposed method:

- The cost of calculation increase exponentially when structures of agents and environments become complicated. In order to reduce the complexity, several techniques like mean field approximation and N-head dynamic programming should be applied to these models.
- The incremental learning will suit to acquire high-level cooperative behaviors. We may be able to realize the step-by-step learning using dependency of the initial parameters.

References

[Bengio and Frasconi, 1995] Yoshua Bengio and Paolo Frasconi. An input output hmm architecuture. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, pages 427–434. The MIT Press, 1995.

[Brand, 1997] Matthew Brand. Coupled hideen markov models for modeling interacting processes. Perceptual Computing/Learning and Common Sense Technical Report 405, MIT Lab, jun 1997.

[Ghahramani and Jordan, 1997] Zoubin Ghahramani and Michael I. Jordan. Factorial hidden markov models. *Machine Learning*, 29:245–275, 1997.

[hidden Markov models for complex action recognition, 199 Coupled hidden Markov models for complex action recognition. Matthew brand and nuria oliver and alex pentland. Perceptual Computing/Learning and Common Sense Technical Report 407, MIT Media Lab, 20 1996.

[Jordan *et al.*, 1997a] Michael I. Jordan, Zoubin Ghahramani, and Lawrence K. Saul. Hidden markov decision trees. In Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, page 501. The MIT Press, 1997.

[Jordan *et al.*, 1997b] Michael I. Jordan, Zoubin Ghahramani, and Lawrence K. Saul. Hidden markov decision trees. In Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems 9*, page 501. The MIT Press, 1997.

[Jordan *et al.*, 1999] Michael I. Jordan, Zoubin Ghahramani, Tommi Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

Real-time Decision Making under Uncertainty of Self-Localization Results

Takeshi FUKASE, Yuichi KOBAYASHI, Ryuichi UEDA and Tamio ARAI
Department of Precision Engineering,
School of Engineering, The University of Tokyo
{fukase, kobayasi, ueda, arai}@prince.pe.u-tokyo.ac.jp

Abstract

In this paper, we present a real-time decision making method for a quadruped robot whose sensor and locomotion have large errors. We make a State-Action Map by off-line planning considering the uncertainty of the robot's location with Dynamic Programming (DP). Using this map, the robot can immediately decide optimal action which minimizes the time to reach a target state at any states. The number of observation is also minimized. We compress this map for implementation with Vector Quantization (VQ). The total loss of optimality through compression is minimized by using the differences of the values between the optimal action and the others.

1 Introduction

In Sony Four-Legged Robot League, self-localization with insufficient sensor information and unreliable locomotion is an important problem[1, 2]. Moreover, to localize itself, the robot must swing its head to look for landmarks because the robot's camera has a narrow visual field. It is required to keep the frequency of this "off-ball" observation behavior as small as possible to cope with the dynamic changing situation. As a result, the robot is required to judge whether it should swing its head to look for landmarks or execute a walking action. The simplest criterion for the judgement is to adapt a fixed threshold of the location's uncertainty [3] , but there are many situations in which the robot can decide its action without precise self-localization results. Thus, the time for taking observation behavior, or the observational cost, is an important factor to deal with uncertainty of the SONY Legged Robot.

Mitsunaga *et al.* proposed a decision making tree that gives consideration to the observational strategy based on information criterion[4]. The tree is made from the large experimental teaching data ,which contain the information of the motion planning and the probability distribution models of sensing and locomotion. In order to apply larger problems, however, the decision making architecture should once analyze these two kinds of information separately.

Our approach to the real-time decision making method deals with:

- modeling uncertainty in the robot's locomotion and observations using uniform distribution;
- adopting Dynamic Programming [5, 6] for motion planning in discrete state space;
- enlarging the state space of planning (configuration space) so as to include the uncertainty parameters; and
- compressing the off-line calculated information using Vector Quantization.

The robot's locomotion models and observation models are taken into consideration respectively in the process of Dynamic Programming ,which guarantees the optimality. By the expansion of the state space to include uncertainty parameters, the observational cost can be computed in the framework of DP. At the same time, the on-line calculation can be reduced by referring the off-line calculated database as far as the memory permits.

From another point of view, an effective design of reflective behavior has been proposed by Hugel *et al.*[7]. They connected sensor information and locomotion sequences so that the robot can push the ball toward the goal effectively. But their design is highly sophisticated just in the sense of the designer's empirical intuition. Our framework realizes the similar behavior as [3, 4, 7], but is based on the automatic design that covers the whole state space. So the idea of the architecture can be applied to the larger problems more easily.

In section 2, the task in the Legged Robot League is specified. Section 3 outlines the proposed real-time motion decision method. In Section 4,5, and 6, the implementation of the method to the task is described. In Section 7 and 8, the proposed method is evaluated in the computer simulation and the experiment, respectively.

2 Task and Assumption

The task is to approach the ball from the proper direction so as not to attack the own goal.

- There are eight discrete walking actions and one observation action.
- The walking actions yield large odometry errors.
- The six unique landmarks are placed around the field.
- The measurement of distance to the landmark contains large errors. A distance is always measured larger than the actual distance because of our image processing algorithm's property [1].
- The robot does not recognize the landmarks while walking towards the ball (it just tracks the ball).
- The relative position of the ball can be measured precisely.
- The robot recognizes the landmarks by the observation behavior, which is to swing its head horizontally 180[deg].

The state of the robot and the ball can be represented by the next five variables, which are shown in Fig.1.

- The robot's pose on the coordinate of the field: (x, y, θ). x-axis is parallel with the touch line, y-axis is parallel with the center line.
- The ball's relative position from the robot: (r, ϕ). r denotes the distance between the robot and the ball. ϕ denotes the direction of the ball from the robot.

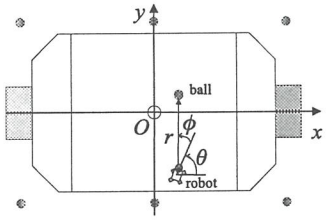


Fig. 1: Five variables for the state's representation.

Without sufficient observation, the precise pose (x, y, θ) of the robot can not be obtained. Therefore the state of the robot should be expressed by probability density functions.

3 Real-Time Decision Making

From the above-mentioned discussion, real-time decision making methods are required to meet following properties:

1 automatic design which can discuss optimality, not based on empirical hand-coding
2 low computational cost
3 ability to express the observational cost and the uncertainty of the pose information

To meet the first characteristic, we adopt Dynamic Programming[5], which is widely used to solve the optimal control problems. The low computational property means that the robot ERS-2100 has 32MB RAM and its calculation speed is equivalent to the 200MHz PC. The Behavior Maps are too large to implement on the robot. In order to compress the Behavior Maps, we apply Vector Quantization (VQ) [8].

The aspects of uncertainty and observational costs are most important in this paper. The uncertainty can be considered as variables of the state space [9].

3.1 Motion Planning with Dynamic Programming

Let $\boldsymbol{x} \in \mathcal{X} \subset R^n$ denote the state vector and $\boldsymbol{u} \in \mathcal{U} \subset R^m$ denote the control input vector. The system dynamics in discrete time is expressed as follows:

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}[\boldsymbol{x}_k, \boldsymbol{u}_k]. \tag{1}$$

The deterministic control policy is given by $\boldsymbol{u}_k = \pi(\boldsymbol{x}_k)$. The purpose of the optimal control problem is to find the optimal policy $\pi^*(\boldsymbol{x})$ that maximizes

$$S = \sum_{k=1}^{T} R[\boldsymbol{x}_k, \boldsymbol{u}_k], \tag{2}$$

where $R[\boldsymbol{x}_k, \boldsymbol{u}_k]$ is the immediate evaluation function of each state and control input pair and T is the time step until the task ends.

In the navigation task, the state vector consists of $\boldsymbol{x} = (x, y, \theta)$. In this research, the state vector includes the ball information. Thus $\boldsymbol{x} = (x, y, \theta, r, \phi)$ has five dimensions. We substitute discrete s and a for $\boldsymbol{x}$ and $\boldsymbol{u}$, respectively. Here, $\mathcal{S}$ and $\mathcal{A}$ are the set of discrete states and actions. The Bellman equation in discrete time and space (without the discount factor) can be formulated as follows:

$$V^*(s) = \max_a \sum_{s'} \mathcal{P}^a_{ss'}[\mathcal{R}^a_{ss'} + V^*(s')], \tag{3}$$
$$Q^*(s, a) = \max_a \sum_{s'} \mathcal{P}^a_{ss'}[\mathcal{R}^a_{ss'} + \max_{a'} Q^*(s', a')], \tag{4}$$

where $\mathcal{P}^a_{ss'}$ denotes the transition probability from state s to s' by taking action a, and $\mathcal{R}^a_{ss'}$ denotes the immediate evaluation given to the state transition from s to s' by taking action a. The optimal state-value function $V^*(s)$ denotes the expected evaluation which is given after state s, by taking actions under the optimal policy π^*. The optimal action-value function $Q^*(s, a)$ denotes the expected evaluation after taking action a at state s, in the same way. This discrete time and space DP approach is applied to the navigation task in Legged Robot League[10].

3.2 Planning Optimal Behavior Under Uncertainty

When the motions are planned, the variance of pose estimation and the observational cost should be taken into consideration. Fig.2 shows an example where the variance of the posture estimation enlarges when the robot executes a walking action. Fig.3 shows an example where the variance decreases when the robot takes observation.

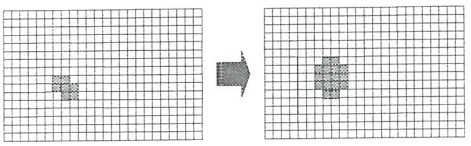


Fig. 2: A state's transition on the occasion of the robot's movement.

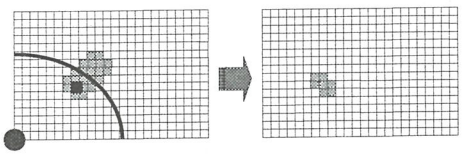


Fig. 3: A state's transition on the occasion of a landmark observation.

These factors can be formulated in the motion planning literature as follows:

$$(\boldsymbol{x}_{k+1}, \boldsymbol{\psi}_{k+1}) = \boldsymbol{f}'[(\boldsymbol{x}_k, \boldsymbol{\psi}_k), (\boldsymbol{u}_k, \boldsymbol{\omega}_k)], \tag{5}$$

where $\boldsymbol{\psi}$ denotes the state variance vector and $\boldsymbol{\omega}$ denotes the observational control vector. Thus, the optimal control problem can be solved in the expanded state space $\{\boldsymbol{x}, \boldsymbol{\psi} | \boldsymbol{x} \in \mathcal{X}, \boldsymbol{\psi} \in \Psi\}$. Fig.4 shows the abstraction of the state transition in the expanded state space $(\boldsymbol{x}, \boldsymbol{\psi})$. The increase of the variance in the original state space can be expressed as the transition along the $\boldsymbol{\psi}$ axis which is indicated in the right hand of the figure.

3.3 Compression of Map with Vector Quantization

The map should be compressed in order to implement on the limited amount of robot's memory. We apply Vector Quantization (VQ) as a data compression method. The map is distorted through compression and the optimality of action data is lost. We should pay attention not to maximizing the decode

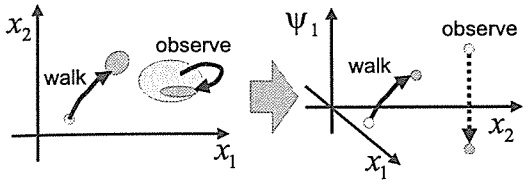


Fig. 4: The state transition in the expanded state space.

Table 1: The way of quantization.

	# of elements	Width of an element
x	28	100[mm]
y	18	100[mm]
θ	24	15[deg]
r	10	100[mm](near) - 600[mm],∞(far)
ϕ	12	15[deg]

rate but to minimizing the increase of the time to reach the target. Hence, we calculate the differences between the optimal action and the others based on the value function, and utilize it as a distortion measure.

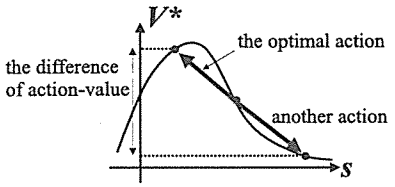


Fig. 5: When the robot takes a non-optimal action, the state-value after the action becomes lower than the state-value after the optimal action. We use this difference for the definition of distortion measure.

4 Implementation 1: Dynamic Programming

4.1 Symbols definition

Firstly, we decide the definition of a state s and a set of all states $\mathcal{S}$. Some of states in $\mathcal{S}$ are the target states. We define the set of them as $\mathcal{S}^*$. We quantize these parameters as shown in Table 1 and Fig.6 so as to quantize the state space.

If we do not consider uncertainty of the self-localization result, a state has five dimensions, and is represented as $s[i_x, i_y, i_\theta, i_r, i_\phi]$ (i_τ means ith index of quantized τ). However, we add one more parameter ψ which denotes the shape of region in which the robot exists with high probability. ψ is quantized to a combination of some $s_{\rm c}[i_x, i_y, i_\theta]$s, which are regarded as cuboids in the (x, y, θ) space (Fig.7). Since a ψ represents only a shape, the area which the robot exists with high probability is represented as $s_{\rm r}[i_x, i_y, i_\theta, i_\psi]$. We restrict the number of ψs to 811 though combinations of cuboids are much more than 811 so as to save the amount of calculation. We define i_ψ so that the larger i_ψ is, the more the number of ψ's cuboids is. Eventually, we let a state $\forall s \in \mathcal{S}$ have six indexes as $s[i_x, i_y, i_\theta, i_r, i_\phi, i_\psi]$. Then, the number of states N is 1177182720, and about 3% of them are in $\mathcal{S}^*$. Hereafter, we often describe $s[i_x, i_y, i_\theta, i_r, i_\phi, i_\psi] \in \mathcal{S}$, $s_{\rm r}[i_x, i_y, i_\theta, i_\psi] \in \mathcal{S}_{\rm r}$ and $s_{\rm b}[i_r, i_\phi] \in \mathcal{S}_{\rm b}$ as s_i, $s_{{\rm r}i}$ and $s_{{\rm b}i}$, respectively.

Secondly, we define some symbols on actions. Our robot has some fixed locomotion actions and a observation action. $\mathcal{A} = \{a_i | i = 1, 2, \ldots, M\}$ denotes the set of these actions. Each action has the following attributes:

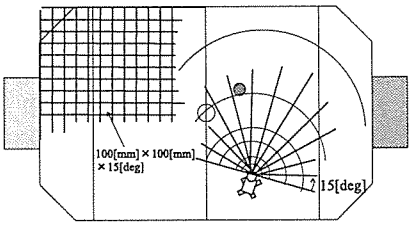


Fig. 6: The robot's position is divided into three-dimensional grids, and the ball's position is divided into two-dimensional grids. The ball's distance is divided irregularly since precise distance information is not important when the ball is far from the robot.

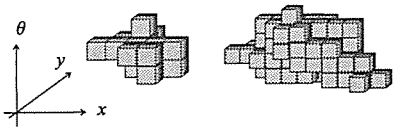


Fig. 7: A quantized ψ consist of three-dimensional cuboids in (x, y, θ) space.

- The time consumption : $\mathcal{R}^{a_i}(< 0)$. We regard a action's time consumption as negative reward. We assume that $\forall \mathcal{R}^{a_i}$ is independent of $\forall s \in \mathcal{S}$.
- Region of the robot's pose after taking a_i at $(x, y, \theta) = (0, 0, 0)$: $M^{a_i}(\bar{x} \pm \delta_x, \bar{y} \pm \delta_y, \bar{\theta} \pm \delta_\theta)$. $(\bar{x}, \bar{y}, \bar{\theta})$ denotes the average and $(\delta_x, \delta_y, \delta_\theta)$ denotes the maximum error. We assume that the probability density of the robot's existence in the M^{a_i} is an uniform distribution. We assume that the transfer amount is independent of the robot's pose. We can calculate $M_p^{a_i}$, which is the region of the robot's pose after a_i at p, from M^{a_i}. $M_p^{a_i}$ is the basis when each transition probability $\mathcal{P}_{ss'}^{a_i}$ is calculated. Table 2 shows these regions.

Table 2: Actions and Parameters of them

Action name	M^{a_i}[mm],[deg]	$\mathcal{R}^{a_i}$ [msec]
1:forward	$(70 \pm 30, 0 \pm 15, 0 \pm 6)$	-768
2:backward	$(-40 \pm 40, 0 \pm 15, 0 \pm 6)$	-768
3:rightside	$(0 \pm 20, -60 \pm 30, 4 \pm 4)$	-896
4:leftside	$(0 \pm 20, 60 \pm 30, -4 \pm 4)$	-896
5:rightforward	$(10 \pm 10, 37.5 \pm 17.5, -14.5 \pm 6.5)$	-832
6:leftforward	$(10 \pm 10, -37.5 \pm 17.5, 14.5 \pm 6.5)$	-832
7:roll_right	$(35 \pm 15, -35 \pm 15, 11.5 \pm 6.5)$	-832
8:roll_left	$(35 \pm 15, -35 \pm 15, -11.5 \pm 6.5)$	-832
9:observation	$(0 \pm 0, 0 \pm 0, 0 \pm 0)$	-2800

Finally, we describe the policy and the state-value function. A policy $\pi(s)$ gives an action when a state s is given. State-Action Map represents the optimal and deterministic policy $\pi^*(s)$. A state-value function $V^\pi(s)$ denotes value of a state s under a policy $\pi(s)$. In our method, it means the expected time to reach one of the target states from the state s. Especially, $V^*(s)$ denotes the state-value function under the optimal policy $\pi^*(s)$.

4.2 The calculation of $\mathcal{P}_{s_i s_j}^{a_k}$

With above symbols, the equation (4) is rewritten as

$$V^*(s_i) = \max_{a_k} \sum_{j=1}^{N} \mathcal{P}_{s_i s_j}^{a_k} [\mathcal{R}^{a_k} + V^*(s_j)] \ (\forall a_k \in \mathcal{A}). \quad (6)$$

Before we explain it, we should refer to the calculation algorithm of $\mathcal{P}^{a_k}_{s_i s_j}$. We can calculate $\mathcal{P}^{a_k}_{s_i s_j}$ by the next two algorithms.

4.2.1 Calculation of Pose's Transition

We do not treat stochastically the renewal of s_{ri} after an action a_k (we represent it as $s^{a_k}_{ri}$) since s_{ri} and $s^{a_k}_{ri}$ are already stochastic in themselves. Therefore, we choose one $s^{a_k}_{ri}$ with the next algorithm.

$k \leftarrow 1$
do
  $\boldsymbol{p} \leftarrow (x_{\rm rnd}, y_{\rm rnd}, \theta_{\rm rnd}) \in s_{ri}$
  $\boldsymbol{q} \leftarrow (x_{\rm rnd}, y_{\rm rnd}, \theta_{\rm rnd}) \in M^{a_j}_{\boldsymbol{p}}$
  $s_{ck} \leftarrow s_{\rm c} \ni \boldsymbol{q}$.
  k^{++}
loop sufficiently
$\hat{s}_{\rm r} \leftarrow \{s_{c\ell} | \ell = 1, 2, \ldots, k\}$
$s_{ri} \leftarrow$ the most proper $s_{\rm r}$ to approximate $\hat{s}_{\rm r}$.

$x_{\rm rnd}, y_{\rm rnd}$ and $\theta_{\rm rnd}$ are continuous random values. It takes 30 minutes to execute this algorithm about all i, j sets by a Pentium III 866 MHz PC.

4.2.2 Calculation of Ball Position's Transition

We define $\mathcal{P}^{a_k}_{s_{bi} s_{bj}}$ as the probability which the ball's position becomes s_{bj} after an action a_k from s_{bi}. $\mathcal{P}^{a_k}_{s_{bi} s_{bj}}$ is calculated by the following algorithm.

$n_\ell \leftarrow 0 \ \ (\ell = 1, 2, \ldots, N)$
do
  $\Delta \boldsymbol{p} \leftarrow (x_{\rm rnd}, y_{\rm rnd}, \theta_{\rm rnd}) \in M^{a_k}$
  $\boldsymbol{b} \leftarrow (r_{\rm rnd}, \phi_{\rm rnd}) \in s_{bj}$
  $\boldsymbol{b}' \leftarrow$ the relative $\boldsymbol{b}$'s variation
    when the robot moves $\Delta \boldsymbol{p}$.
  $n_\ell{}^{++}$ if $s_{b\ell} \ni \boldsymbol{b}'$.
loop sufficiently
$\alpha \leftarrow \sum_{\ell=1}^{N} n_\ell$
$\mathcal{P}^{a_k}_{s_{bi} s_{bj}} \leftarrow n_j / \alpha$

This algorithm does not take more than some seconds by the previous PC. Eventually, we can obtain $\mathcal{P}^{a_k}_{s_i s_j}$ as:

$$\mathcal{P}^{a_k}_{s_i s_j} = \begin{cases} 0 & \text{if } s^{a_k}_{ri} \neq s_{rj} \\ \mathcal{P}^{a_k}_{s_{bi} s_{bj}} & \text{if } s^{a_k}_{ri} = s_{rj} \end{cases} \tag{7}$$

from these algorithms.

4.3 DP algorithm

We use the value iteration algorithm [6] to obtain the approximation of the optimal policy π^*. The value iteration is represented as the following algorithm.

$V(\boldsymbol{s}) = 0$ (for all $\boldsymbol{s}$).
Dim ϵ as a small positive threshold
do
  $\Delta \leftarrow 0$
  for $i = 1$ to N
    if $s_i \notin \mathcal{S}^*$
      $v \leftarrow V(s_i)$
      $V(s_i) \leftarrow \max_{a_k} \sum_{j=1}^{N} \mathcal{P}^{a_k}_{s_i s_j} [\mathcal{R}^{a_k} + V(s_j)]$
      $\Delta \leftarrow \max\{\Delta, |v - V(s_i)|\}$
    end if
  next
loop until $\Delta < \epsilon$
$\pi^*(s_i) \leftarrow \arg\max_{a_k} \sum_{j=1}^{N} \mathcal{P}^{a_k}_{s_i s_j} [\mathcal{R}^{a_k} + V(s_j)]$
$(i = 1, 2, \ldots, N)$

This algorithm takes about two days.

5 Implementation 2: Compression of State-Action Map based on Vector Quantization

We use Vector Quantization (VQ) [8] so as to compress the State-Action Map π^*, since the data amount of π^* is about 500 MB and the robot does not have such a huge amount of RAM. The maximum volume of data that can be transferred to the robot is 16MB.

5.1 Definition of Vector

We explain the way representative vectors are made. At first, we divide the map since the State-Action Map is too large to be executed VQ all at once. We prepare sets of states $\mathcal{S}'_j = \{\boldsymbol{s}[i_x, i_y, i_\theta, i_r, i_\phi, i_\psi] | i_\psi = 2j - 1, 2j\}$. VQ is executed in each $\mathcal{S}'_j$ independently. Next, we decompose each $\mathcal{S}'_j$ into ordered sets, i.e. $\mathcal{S}'_j = \{\mathcal{S}''_{jk} | k = 1, 2, \ldots, N_k\}$. Each of them corresponds to one vector for VQ. We define $\mathcal{S}''_{jk}$s as six-dimensional cuboids (Fig. 8), i.e. $\mathcal{S}''_{jk} = \{\boldsymbol{s}[i_x, i_y, i_\theta, i_r, i_\phi, i_\psi] | k^{(j)}_{\alpha 1} \leq i_\alpha \leq k^{(j)}_{\alpha 2} \ (\alpha = 1, 2, \ldots, 6. \ i_1 = i_x, i_2 = i_y, \ldots, i_6 = i_\psi.)\}$ $(\forall k, k', \alpha \rightarrow k^{(j)}_{\alpha 2} - k^{(j)}_{\alpha 1} = k'^{(j)}_{\alpha 2} - k'^{(j)}_{\alpha 1})$. If we define ℓ as

$$\ell = \sum_{\alpha=2}^{6} (i_\alpha - k^j_{\alpha 1}) \prod_{\beta=1}^{\alpha-1} \omega_\beta + i_1 \ (\omega_\beta = k^j_{\beta 2} - k^j_{\beta 1}) \tag{8}$$

and represent $\mathcal{S}''_{jk} = \{\boldsymbol{s}^{(jk)}_\ell | \ell = 1, 2, \ldots, N_\ell\}$, we can regard $\boldsymbol{v}_{jk} = (\pi^*(\boldsymbol{s}^{(jk)}_1), \pi^*(\boldsymbol{s}^{(jk)}_2), \ldots, \pi^*(\boldsymbol{s}^{(jk)}_{N_\ell}))$ as a vector which is used in VQ.

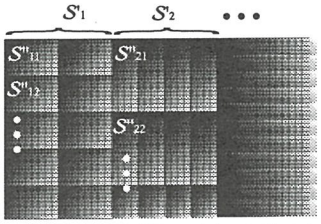


Fig. 8: Model of the states' division.

The decision way of ω_α $(\alpha = 1, 2, \ldots, 6)$ should satisfy that the same vectors are produced in $\mathcal{S}'_j$ as much as possible, since it is favorable for VQ. We decide ω_α to minimize the next entropy function:

$$-\sum_{k=1}^{N_k} \frac{1}{N_k} \log \frac{N_{\rm s}(\boldsymbol{v}_{jk})}{N_k}, \tag{9}$$

where $N_{\rm s}(\boldsymbol{v}_{jk})$ means the number of elements which are the same with $\boldsymbol{v}_{jk}$ (it counts $(\boldsymbol{v}_{jk})$). In our case, each $\mathcal{S}'_j$ has 2903040 states except for $\mathcal{S}'_{406}$[1]. We decide $N_\ell = 72$ and $N_k = 40320$. And we determine the appropriate ω_α $(\alpha = 1, 2, \ldots, 6)$ respectively as for $\mathcal{S}'_j$ $(j = 1, 2, \ldots, 405)$.

[1] All states in $\mathcal{S}'_{406}$ represent the state which the robot does not know its pose. Therefore when the robot is a state in $\mathcal{S}'_{406}$, the robot can do nothing but a landmark observation. Hence, we do not need DP for $\mathcal{S}'_{406}$.

5.2 Definition of Distorsion

Next, we must define distorsion of any two vectors in the vector space $\mathcal{V}_j$ which the vectors $\boldsymbol{v}_{jk}$ $(k = 1, 2, \ldots, N_k)$ belong to. We define the distorsion between $\forall \boldsymbol{v} \in \mathcal{V}_j$ and $\forall \boldsymbol{w} \in \mathcal{V}_j$ as

$$D[\boldsymbol{v}, \boldsymbol{w}] = \sum_{\ell=1}^{N_\ell} D[v_\ell, w_\ell], \tag{10}$$

where $v_\ell, w_\ell \in \mathcal{A}$ are the ℓth elements of $\boldsymbol{v}$ and $\boldsymbol{w}$ respectively. Therefore, we must define the distorsion $D[a_m, a_n]$ about $\forall m, n$. $D[a_m, a_n]$ is calculated from the optimal action-value function (4) as

$$D[a_m, a_n] = \frac{\sum_{k=1}^{N_k} \sum_{\ell=1}^{N_\ell} \delta[\pi^*(\boldsymbol{s}_\ell^{(jk)}), a_m](Q^*(\boldsymbol{s}_\ell^{(jk)}, a_m) - Q^*(\boldsymbol{s}_\ell^{(jk)}, a_n))}{\sum_{k=1}^{N_k} \sum_{\ell=1}^{N_\ell} \delta[\pi^*(\boldsymbol{s}_\ell^{(jk)}), a_m]}, \tag{11}$$

where, $\delta[\alpha, \beta]$ is Kronecher's Delta.

5.3 Execution of VQ

We use Pairwise Nearest Neighbor (PNN) algorithm [11] to choose an initial codebook $\mathcal{C}_j$, which belongs to $\mathcal{S}'_j$. Each $\mathcal{C}_j$ contains representative vectors $\boldsymbol{c}_{ji}$ $(i = 1, 2, \ldots, N_c)$. Each representative vector has the database of vectors $\boldsymbol{v}_{jk}$ which are associated with it. We define sets of vectors which are associated with a vector as the cluster $R_i = \{\boldsymbol{v}_n | n = 1, 2, \ldots, N_{R_i}\}$, and the represented vector as:

$$\boldsymbol{c}_i = \arg\min_{\boldsymbol{w}} \sum_{k=1}^{N_{R_i}} D[\boldsymbol{w}, \boldsymbol{v}_k] \ (\boldsymbol{w} \in \mathcal{V}_j). \tag{12}$$

Moreover, We define the distorsion-measure of R_i as:

$$d_i = \sum_{k=1}^{N_{R_i}} D[\boldsymbol{c}_i, \boldsymbol{v}_k]. \tag{13}$$

PNN makes the number of representative vectors reduce from the number of vectors in $\mathcal{S}'_j$ to N_c as the following way.

$R_k = \{\boldsymbol{v}_{jk}\}$ $(k = 1, 2, \ldots, N_k)$, $n \leftarrow N_k$
do
  $(i, j) \leftarrow \arg\min_{(\alpha,\beta)}(d_{\alpha\beta} - d_\alpha - d_\beta)$
    $(R_{ij} = R_i \cup R_j,\ R_i \neq \emptyset,\ R_j \neq \emptyset)$
  $R_i \leftarrow R_{ij}$
  $R_j \leftarrow \emptyset$
  n^{--}
until $n = N_c$

Here we improve these initial codebooks choosen by PNN. We adopt Generalized Lloyd Algorithm (GLA) which is a decent algorithm to refine a codebook iteratively. GLA minimizes the distortion as a result of iteration. We apply this algorithm until the update of the codebooks does not happen.

6 Implementation 3: The on-line algorithm

The tasks of the on-line part are to recognize the current state of the robot and to search an optimal action from the codebooks. We use Uniform Monte Carlo Localization (Uniform MCL) [1] for self-localization, and for modeling of state transitions which are caused by the landmark observation action. Simulations of Section 7 use this state transition models. In experiments of Section 8, the robot specifies the current state of the robot s_{ri} with Uniform MCL results. Uniform MCL approximates the possible region in which the robot exists with a lot of points in the space (x, y, θ). When the points crowd in narrow region, it means that the robot knows its pose precisely. And the points are scattered all over the space, it means that the robot does not identify its pose. We use the points immediately to specify the current state at the experiments.

After the former part specifies s_x, the latter part chooses an appropriate action $\hat{a}$ as following procedure.

1 Change the subscript from s_x to $s_{\ell_x}^{(j_x k_x)}$.
2 Choose a representative vector $\boldsymbol{c}_{j_x i_x}$ from $\mathcal{C}_{j_x}$ by k_x.
3 $\hat{a} \leftarrow$ the ℓ_xth element of $\boldsymbol{c}_{j_x i_x}$

7 Simulation

7.1 Purpose and Conditions

In this section, we inspect the efficiency of our method by simulation. Especially, the results of following two cases about judging whether to observe or not are compared in order to verify the effectiveness of considering the self-localization's uncertainty. We compare following two cases.

1 Referring the map case: Compressed map is utilized for all the decision making including the judgment of observation.
2 Threshold case: Fixed threshold for judgment is settled on the width of the probability distribution of the robot's pose. The Compressed map without variance is used only for choosing an optimal walking command.

The thresholds for judgment were fixed at $(x_{\text{th}}, y_{\text{th}}, \theta_{\text{th}}) = (600[\text{mm}], 500[\text{mm}], 60[\text{deg}])$. Other conditions are settled as follows.

- Initial conditions: Table 3 shows the initial positions of the robot and the ball. The robot knows the initial positions completely.
- Locomotion: The robot's real pose is updated with random error shown in Table 2. The absolute position of the ball on the soccer field is fixed and the robot always knows the relative position to the ball with no error. The robot updates its estimating state according to the transition probability.
- Observation: The robot acquires the relative position to landmarks with random errors.
- Terminative condition: In referring the map case, the task is terminated when the robot's estimating state belongs to the terminative states. In using threshold case, it is terminated when the robot's estimating pose, supposing that the estimating pose has no uncertainty, belongs to the terminative states.
- Definition of the success cases: The robot actually reaches a target position.

Table 3: Initial positions of the robot and the ball.

	r[mm]	ϕ[deg]	x[mm]	y[mm]	θ[deg]
Sim.1	2100	30	-1000	-600	0
Sim.2	1800	0	1000	0	180
Sim.3	2100	60	1000	-600	90

7.2 Results and Discussion

We simulated 10 times on each case and initial conditions. The results are shown in Table 4.

In the referring the map case, the robot succeeded to reach the target position at all trials. This indicates that the calculation of DP converged and the distortion of the compressed

map was small enough. In the using the thresholds case, there were some failures. This means that the robot should observe much more. In the referring the map case, both the average number of observation and the time to reach the target were smaller than using thresholds case. This means that there were some cases that a observation was unnecessary and the robot should walk in spite of the large uncertainty of the robot's pose. These results indicate that the robot observed more effectively as a result of the off-line planning, which considers the uncertainty of its pose, and that the robot could reduce the time consumption to reach the target pose.

Table 4: The results of the simulations.

	Referring the map case		
	Time[sec]	**# of observation**	**Success rate**
Sim.1	31.4	2.6	10/10
Sim.2	37.6	3.6	10/10
Sim.3	35.9	2.8	10/10
	Threshold case		
	Time[sec]	**# of observation**	**Success rate**
Sim.1	34.4	4.0	10/10
Sim.2	41.5	4.6	8/10
Sim.3	38.6	3.6	9/10

8 Experiment

We implemented the described method and evaluated it by experiments. The initial conditions are settled on the same as the simulation. We judged success or failure according to the position when the robot touched the ball at first time.

The results of the experiments are shown in Table 5. Note that the average number of observation is smaller than that of simulation. This is due to the quantization of the probabilistic distribution of robot's pose. The expansion of distribution was delayed because Uniform MCL has superior ability to represent the uncertainty of robot's pose.

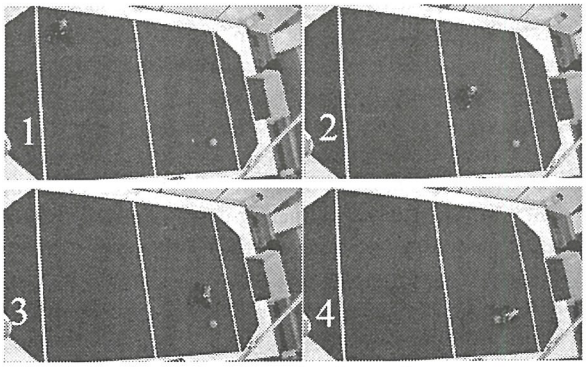


Fig. 9: An example of experiment.

Table 5: The results of the experiment.

Trial	**Time[sec]**	**# of observation**	**Success rate**
1	27.5	1.4	9/10
2	42.0	3.0	6/10
3	41.1	2.3	8/10

The total success rate was about 75%. The failure cases were that the robot touched the ball unintentionally. These were due to the mismeasurement of the ball, which was not assumed in our model. One supposable approach to improve the success rate is to calculate the transition probability of the state that the ball is situated near the robot experimentally and utilize it in DP calculation.

9 Conclusion and Future works

We took the uncertainty of the robot's pose into account by expanding the state space and designed a State-Action Map with DP by off-line calculation. The map was compressed with VQ in order to implement on the limited amount of robot's memory. We also defined the distorsion between any two actions based on the action-value function. The total distortion of the map through compression was minimized as a result. By the simulations and experiments, it was verified that the robot observes the landmarks more efficiently compared with the fixed threshold case.

In this paper, we calculated the average of the difference of action-value function in order to save the amount of calculation of VQ. However, the difference of action-value function depends on the state and the distorsion between two actions should be calculated at each state. Therefore, the distortion will be improved by directly utilizing the value of the action-value function and by updating the value function occasionally while VQ calculation.

References

[1] R. Ueda, T. Fukase, Y. Kobayashi, T. Arai, H. Yuasa and J. Ota: "Uniform Monte Carlo Localization – Fast and Robust Self-localization Method for Mobile Robots," Proc. of ICRA-2002, to appear.

[2] S.Lensor and M.Veloso: "Sensor resetting localization for Poorly Modeled Mobile Robot," Proc. of ICRA-2000, pp.1225-1232, 2000.

[3] E. Winner and M. Veloso: "Multi-Fidelity Robotic Behaviors: Acting With Variable State Information," Proc. of the Seventeenth National Conference on Artificial Intelligence, Austin, August 2000.

[4] N. Mitsunaga and M. Asada: "Observation strategy for decision making based on information criterion," Proc. of IROS-2000, pp.1211-1216, 2000.

[5] R. Bellman, S. Dreyfus: "Applied Dynamic Programming," Princeton University Press, 1962.

[6] R. S. Sutton and A. G. Barto: "Reinforcement Learning: An Introduction," The MIT Press, 1998.

[7] V. Hugel, P. Bonnin and P. Blazevic: "Reactive and Adaptive Control Architecture Designed for the Sony Legged Robots League in RoboCup 1999," Proc. of IROS-2000, 2000.

[8] A. Gersho and R. M. Gray: "Vector Quantization and Signal Compression," Kluwer Academic Publishers, 1992.

[9] S. M. LaValle: "Roboto Motion Planning: A Game–Theoretic Foundation," Algorithmica, Vol. 26, pp.430-465, 2000.

[10] T. Fukase, M. Yokoi, Y. Kobayashi, R. Ueda: "Quadruped Robot Navigation Considering the Observational Cost," The RoboCup 2001 International Symposium, Seattle, USA.

[11] W. H. Equitz: "A new vector quantization picture coding," IEEE Trans. Acoust. Speech Signal Process., pp. 1568-1575, October 1989.

[12] A. R. Cassandra, L. P. Kaelbling and J. A. Kurien: "Acting under Uncertainty: Discrete Bayesian Models for Mobile–Robot Navigation," Proc. of IROS-96, 1996.

ロボット協調動作としてのモールプレーの検討

Preliminary Implementation of Maul play as a cooperation of Robots

折尾紘幸、大浦和浩、長坂保典、藤吉弘宣、藤井隆司、高橋友一

Hiroyuki Orio, Kazurio Ooura, Yasunori Nagasaka, Hironobu Fujiyoshi, Takashi Fujii, Tomoichi Takahashi

中部大学

Chubu University

ttaka@isc.chubu.ac.jp

Abstract

Specific mechanical devices have made pass plays popular in RoboCup small robot league. Pass plays make one game style, while dribble plays make different game style. This paper describes preliminary experiments of maul play. Multiple robots are cooperated to do maul and to carry a ball. Strategic view of maul play is also discussed.

1 Introduction

Pass a ball to a robot in an open space is an effective strategy in soccer games. In Robot soccer, pass play as one of cooperations among agents/robots have clearly observed at first in games of simulation league. In small size league, top teams showed sometimes passes between robots at 1999. From 2000, spectators recognize them clearly as passes play between robots.

One of reasons that pass play has become common in small size league is that specific mechanical devices are implemented. The devices makes ball handling such as dribble, pass and shoot, easier than ball control by bumping itself against a ball.

One of our team's ideas is robots (even without special devices) carrying a ball by cooperating themselves. Fig. 1 shows the image of the cooperation play of three robots. Robots carry a ball with holding the ball in the center of them. The robots move keeping the formation, as a result of the formation, it blocks the ball from opponents' attack. The play looks like maul play in rugby not soccer [1].

In section2, preliminary experiments to implement maul play by robots are discussed. Section 3 describes

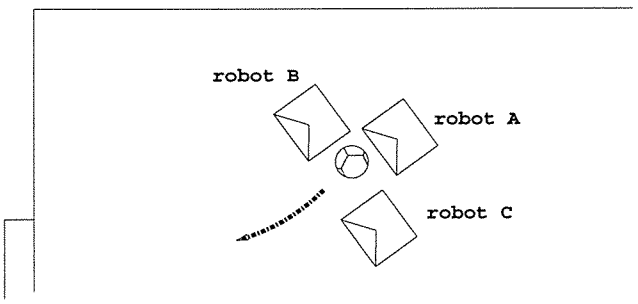


Figure 1: image of maul play by three robots

introducing maul play changes games styles.

2 Robot motion in formation

2.1 general schema of robot control

Using global vision system, each robot are controlled by repeating the following steps during games.

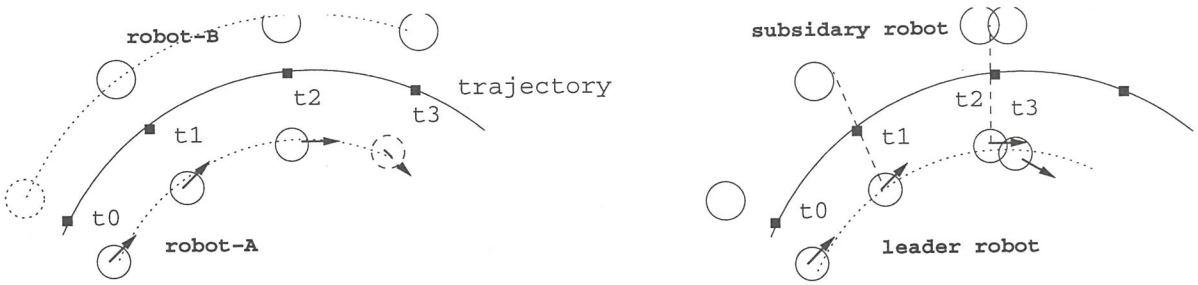
Step 1: Sense objects on the field. The sensed data at t_i are a set of positions of a ball, teammate robots, opponent robots and direction of teammates –

$$\{ \quad (ball\_x_i, ball\_y_i), \\ \{(x_i^k, y_i^k, \overrightarrow{dir}_i^k), k \in (1, \ldots, 5)\}, \\ \{(X_i^l, Y_i^l), l \in (1, \ldots, 5)\}\}$$

where k is indicates numbers of team mate robots and l shows opponent team's.

Step 2: Plan teammate robots' motion at next step and calculate the position and direction of teammate robots at next time t_{i+1}–
$\{pre{-}x_{i+1}^k, pre{-}y_{i+1}^k, \vec{direction}_{i+1}^k), k \in (1, \ldots, 5)\}$.

Step 3: Calculate robot control commands (parameters of motors, or velocity of robots ,..etc),



(a) method 1 (b) method 2

Figure 2: methods of position calculation

Step 4: Send them to robots.

2.2 comparison of position calculation methods

Robots' movements in a form are demonstrated in Cornell Univ.'s team [2]. They move each other and pass a ball with keeping specified formations such as circle formation.

The purpose of our robot control is to carry a ball. We compare two methods to calculate positions of robots at next steps (Fig. 2). The middle solid line in the figure shows a planned trajectory of the midpoint of robots. The dots on the trajectory indicate the point at t_i.

method 1: Positions and directions of robots are calculated independently from the trajectory.

method 2: Position and direction of a robot (the robot is called a leader) are calculated at first. The other robot's position and direction are calculated from the leader robot.

The two methods are equivalent in a simulated world, however, they give different results in a real world where a robot may not move as commands, or the images may not be grabbed clearly. The right figure in Fig. 2 shows a case that the leader robot is stacked at t_2. Then the other robot must stay there to hold a ball between them and not go as in the left figure.

We did two experiments:

experiment 1: Two robots move from a goal to the other goal keeping the space between each other equal.

experiment 2: Two robots move in a circle which center is the center of the field. Fig. 3 shows the initial layout of robots.

The robots which size is 10cm square are controlled based on images grabbed by a global camera. The image

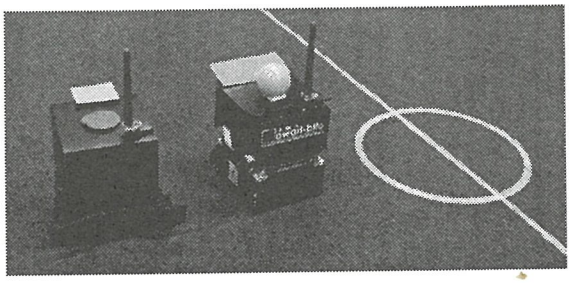


Figure 3: initial layout of robot

size is 480 × 640 dots and the length between dots is about 5 mm. Our vision system processes the images at 30 frames per a second. The robot's speed is changed from 20 cm/s to 60 cm/s. The command are sent to robots per 20 times per a second.

Table. 1 shows the result of robots' movements about a few seconds. The periods are equal for method 1 and method 2 , while are different for speeds condition. The left figure in each column shows the mean distance between robots or between a robot and trajectory. The right figure shows the standard deviation of them. It says that method 1 is better that method 2 at low speed, and they become equal at high speed.

2.3 discussion

In order to move three robots as shown in Fig. 1 and not to drop a ball behind them, the gap between robot B and C should be less than the size of robot A, and robot A should follow robot B and C within the size of a ball.

In a case of 20cm/s speed, the displacement per a command transmission is 1cm which correspond 2 dots. The displacement is also the same order as the deviation in Table. 1.

These consideration leads that an image grabbing de-

Table 1: Distance between robots or between robot and trajectory

		Experiment 1						Experiment 2					
speed		20cm/s		40cm/s		60cm/s		20cm/s		40cm/s		60cm/s	
method 1	robot-robot	20.80	3.78	26.15	7.35	25.77	3.26	20.21	2.70	18.77	2.26	16.47	2.95
	robot1-tra.	10.39	1.86	13.29	3.97	12.94	1.66	10.10	1.33	9.51	1.16	9.55	0.59
	robot2-tra.	10.53	1.98	13.65	4.05	13.01	1.62	10.19	1.46	9.66	1.38	11.22	4.06
method 2	robot-robot	22.32	1.47	24.79	3.78	25.22	4.88	27.58	4.07	17.03	2.48	16.20	2.95
	robot1-tra.	10.10	0.14	10.16	0.24	10.54	0.72	10.13	0.19	9.74	0.37	9.55	0.58
	robot2-tra.	12.57	1.23	15.81	3.58	15.91	4.53	17.68	4.09	8.61	2.16	11.47	4.00

unit [cm] left is mean, right is standard deviation.

vice with more than 480 × 640 dots and more frequent communication are required to implement maul play.

3 Using maul play as one of team strategies

Although the maul play requires precise control of robots to carry the ball to a goal, it blocks the ball from opponent's attack and holds it safely. This provides a new and attractive aspect on planning strategy.

3.1 algorithm of forming maul formation

A basic algorithm to form maul formation consists of

1. rank robots with distance from the ball,
2. controls robots to positions shown in Fig. 1 according to distance rank.

In forming the formation, it is desirable that robot A comes first from own goal side not to shot own goal. And the other robot B and C gather around robot A and follow it.

3.2 implementation of maul formation as strategy

At present, our team strategy are described as combination of a set of rules and potential field method [3]. Maul play can be used as two ways to attack to the opponent goal and defend by holding the ball. It seems to be difficult to represent two aspects, attack and defence, as simple sum of each aspect's field.

4 Conclusion

Maul play is one of cooperation plays of robots, and gives a birth a new play style of games. Preliminary experiments shows that implementing maul play requires to control robot with more precision than usual.

We are now tuning the robot control parameters and developing algorithms how to form a maul formation and make use of it at games.

参考文献

[1] `http://www.rugby365.com/stories/laws/law_book/LAW_000812_19228.shtml`

[2] R. D'Andrea,*et al.*; The Cornell Roboucp Team, in *RoboCup 2000: Robot Soccer World Cup IV*, Lecture Notes in Computer Science, Springer, 2001, pp. 41-51

[3] Y. Nagasaka,*et al.*; Potential Field Approach to Short Term Action Planning in RoboCup F180 League, in *RoboCup 2000: Robot Soccer World Cup IV*, Lecture Notes in Computer Science, Springer, 2001, pp. 345-350

Fast image processing and flexible path generation system for RoboCup small size league

Shinya HIBINO†, Yukiharu KODAMA†, Yasunori NAGASAKA‡,
Tomoichi TAKAHASHI‡, Kazuhito MURAKAMI† and Tadashi NARUSE†
†Aichi Prefectural University, Nagakute-cho, Aichi, 480-1198 JAPAN
‡Chubu University, Kasugai, Aichi, 487-8501 JAPAN

Abstract

In the physical multi-robot systems in RoboCup, it is very important to raise the robustness of vision system, and to give an optimal feedback in order to control a robot to reach the goal point and generate the action under the team strategy. In this paper, we propose two new methods. One is a fast image processing method, which is coped with the spatial variance of color parameters in the field, to extract the positions of robots and ball in 1/30sec. In the labeling algorithm in this method, the separation problem in the interlace format image is solved. Another one is a path generation method in which the robot approaches the goal by changing its direction convergently. By using these two algorithms, the real time processing system which can generate a stable path under a low quality input image is realized.

1 Introduction

In the physical multi-robot systems in RoboCup[1],[2], it is very important to raise the robustness of vision system and to give an optimal feedback in order to control a robot to reach the goal point. From the viewpoints of image processing, there are some technical problems to be solved; for example, (1)since a golf-ball used in small-size league is sphere with dimples, color parameters, r-, g-, b-values, are not constant by shading in the region of the ball in the image, and (2)lighting condition is not the same even in a field of the game, furthermore, (3)it becomes difficult to extract a moving object because the object would appears as separated split patterns in an interlace format image.

On the other hand, from the viewpoints of processing time to control robots, (1)calculation time should be short because all objects such as a ball and robots move at high speed, and (2)a small size input image is desirable, however, the quality of image is getting low.

In this paper, we propose two new methods. One is a fast image processing method, which is coped with the spatial variance of color parameters in the field, to extract the positions of robots and ball in 1/30sec. In the labeling algorithm in this method, the separation problem in the interlace format image is solved. Another one is a path generation method in which the robot approaches the goal by changing its direction convergently. By using these two algorithms, the real time processing system which can generate a stable path under a low quality input images is realized.

In the sections 2 and 3, we explain new image processing method and labeling algorithm, and path generation algorithm with some experimental results, respectively.

2 Image Processing System

In the samll size robot league, an image obtained from a camera installed on ceiling is usually processed to get the position and the direction of each robot and the position of a ball. We also use such image. Since we use an off-the-shelf camera and frame grabber, it is necessary to develop an image processing algorithm which works well for low quality images. In this section, we discuss such an algorithm. Throughout this section, we use a word 'ID' for the identification number of each robot and a word 'object' for an object region such as a ball, color markers and team markers in the image.

2.1 System configuration

We show a configuration of our image processing system in figure 1. We use the Video4Linux as a driver of frame grabber. First, the frame grabber gives the yuv-image (YUV422). Then, search range is calculated by using history information. For the search range, the segmentation algorithm is applied in order to extract the segments of the object, where the segment is a connected component of pixels which have the color value of specified range. The extracted segments are merged to make a candidate of object by using a labeling algorithm. Finally, for the candidates of the object, the size and the location in the image are tested to identify whether it is the true object or not.

Since we use an off-the-shelf camera and frame grabber, it is difficult to get sharp images. An example of

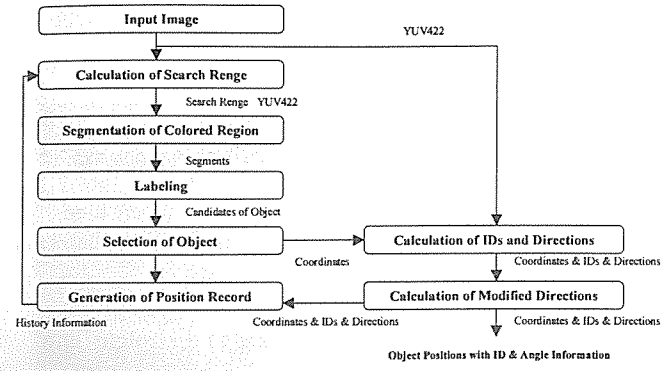


Figure 1: Image processing system

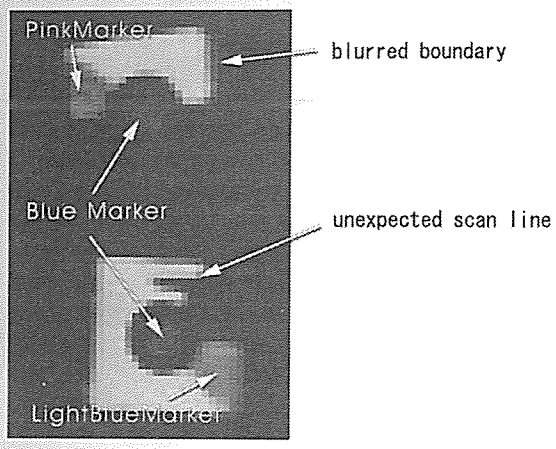


Figure 2: Example of grabbed image

grabbed image is shown in figure 2. In the figure, a boundary of the object area is blurred and an unexpected scan line is contained. For such an image, the above system should work well. To make a system be robust, we developed a new labeling algorithm which is discussed in sec. 2.3

2.2 Color image segmentation for object extraction

In our system, we should detect 6 colors to identify objects, i.e. 1 color for ball (C_1), 2 for team markers(C_2,C_3), and 3 for ID markers (C_4,C_5,C_6). In the color segmentation process, the system classifies each pixel into one of 6 color clusters $C_i(i = 1, 2, ..., 6)$ or the other.

This process utilizes a color image segmentation algorithm developed by CMU[3]. In the algorithm, 32 different colors can be segmented simultaneously. Thus, we assigned 5 different colors for 1 color cluster, which can absorb the subtle color variations of the object caused by the lighting condition. In this way, the image is segmented.

2.3 Labeling algorithm for object extraction

Next step is the labeling process for the segmented image. By this process, a candidate of object is extracted. Since robots and ball move at high speed, conventional labeling algorithm or method doesn't work well for interlace format image. Even though there is a method which processes either of even or odd field of an image, it sacrifices the resolution. Independent processing for even and odd fields makes it difficult to unify the results. Therefore, we developed a new labeling algorithm,– diagonal alternate spread labeling –, which can cope with the interlace format image which has blurred objects of moving robots. We show a summarized algorithm here. In the following, i, k denote the scanning parameter for the x coordinate and j, l for y coordinate. For the symplicity of explanation, it is assumed that the image is binary and an object is black.

2.3.1 Algorithm

diagonal alternate spread labeling

Step0 Let A and B be a set of pixels, respectively, and put $A = \phi, B = \phi$. Let num be a label number, and put $num = 0$.

Step1 Scan the image. If a black pixel (i, j) which is not labeled is found, put it into the set A.

Step2 Do the following.

1) For a pixel (i, j) in the set A, if it is not labeled, label it with num. Then, search the following 8 pixels.

$$(i+2, j+2), (i+2, j-2), (i-2, j+2), (i-2, j-2),$$

$$(i+1, j+1), (i+1, j-1), (i-1, j+1), (i-1, j-1)$$

For each pixel, if it is black, put it into the set B.

2) For a pixel (k, l) in the set B, if it is not labeled, label it with num. Then, search the following 4 pixels.

$$(k, l+1), (k, l-1), (k-1, l), (k+1, l)$$

For each pixel, if it is black, put it into the set A.

3) Repeat *Step2* while new black pixels are gotten.

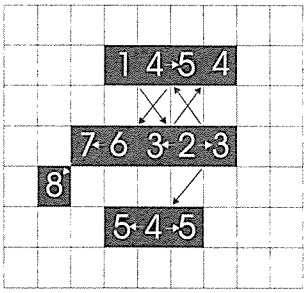
Step3 Increment num and repeat *Step1* and *2* while there are unlabeled black pixels.

We show a labeling example in figure 3.

2.3.2 Effectiveness of the proposed labeling algorithm

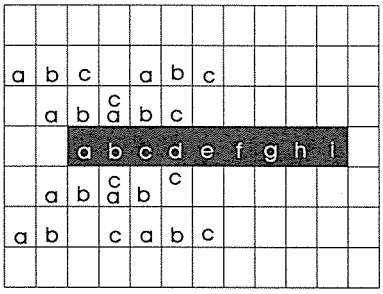
This algorithm can detect line segments, where each of them appears by every two lines, as one object as shown in figure 3. This solves the interlace problem of alternate appearance of black and white lines when the object moves. Using this algrithm, a 4 cm ball with the maximum speed of 240 cm/sec can be detected. If the ball moves over the maximum speed, it is completely separated into 2 objects in the image.

This algorithm has another unique characteristics that a line segment of length n and width 1 (pixel) would be n pieces of independent objects of size 1 (pixel), because



This is an example of segments recognized as one object. The number added to the pixel is a processing order in Step2 of the labeling algorithm. Since the scan is sequential, the first pixel of an object is always the upper left pixel which is numbered 1.

Figure 3: An example of labeling



This is an example of object that each pixel is recognized as a different object. Black pixels are a candidate of object, however, each pixel is labeled with different alphabet (a,b,...,i). White pixels with alphabet show that they are searched in step 2 in conjunction with the black pixel with the same alphabet. (Labels d,..,i is omitted here.)

Figure 4: An example of labeling to a straight line

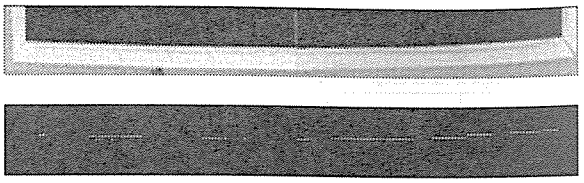
the search to the horizontal and vertical directions occurs after the search to the diagonal directions in Step 2. An example is shown in figure 4. This algorithm works well for a blurred boundary of object shown in figure 5, since the system would delete small size(pixel size) objects as noises by a simple thresholding.

2.4 ID recognition

2.4.1 ID recognition method

In our system, the image processing module recognizes the ID and the direction of each robot. Each robot has a black and white plate on its top as shown in figure 6. In the image, the plate region is detected as an object by the image processing module. By measuring the sector of white region, the ID of robot can be decided. To do so, we prepare the reference table. The size of the table is 40 or 44 entries[1] , that is suited for the size of the

[1] The size of a table is changed according to the situation, because the field size in the image changes with the camera arrangement in the hall.



As a result of the characteristics of frame grabber, blue color appears near a field boundary. The color segmentation algorithm detects it as a candidate segment.

Figure 5: An example of labeling to a straight line

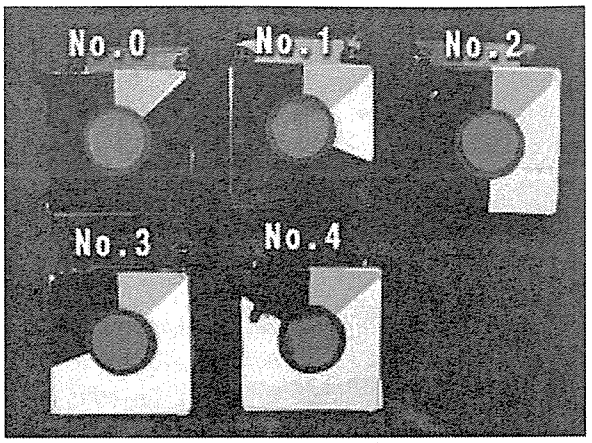


Figure 6: ID plate

robot in an image. The reference table consists of entry number, angle, x and y distance from the center of the plate as shown in table 1.

This table is applied to the plate object as shown in figure 7. The image processing module detects a center of the object and tests the value of pixels which are pointed by the table entries. The module saves the entry number corresponding to the point that the pixel value changes from white to black(it is No.8 in fig.7), and saves the number of pixels whose color is white. In addition, RoboCup rule allowes ID plate to attach other color markers other than black and white, we have set up all markers other than black which is recognized as white by thresholding. It is not difficult to decide this threshold.

This processing is applied to each robot. The ID is obtained by counting the number of pixels judged to be white, and the direction of the robot is given by the table entry corresponding to the pixel whose value changes from black to white.

2.4.2 Resolution and Verification

Since the reference points are arranged on the circle, the ID decision does not depend on the direction of the robot. The angle resolution is about 8 degree(=360/44). And this processing is operated only once to each robot, so the processing time is negligible.

Table 2 shows the accuracy of direction calculation.

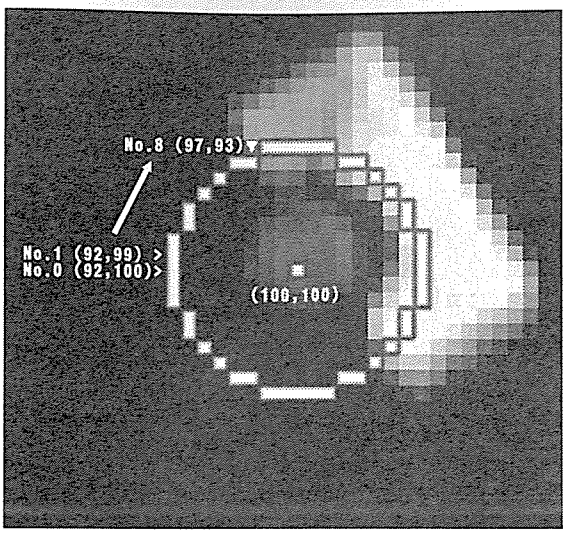


Figure 7: Applying the reference table to the robot object

Table 1: ID reference table
First 11 entries of the 44 entries table are shown.

Entry No.	Angle	X distance	Y distance
0	0	-8	0
1	7	-8	-1
2	14	-8	-2
3	24	-7	-3
4	30	-7	-4
5	38	-6	-5
6	50	-5	-6
7	58	-4	-7
8	65	-3	-7
9	75	-2	-8
10	83	-1	-8

Table 2: Calculated direction

Calculated direction	freq.
172	165
180	731
188	96
196	8

Measured 1000 times.

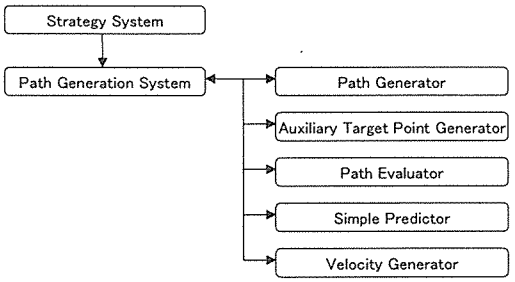


Figure 8: Path generation system

In the experiment, a robot was placed in the horizontal direction and the angle was measured 1000 times. From the table, it is clear that 73% of the calculation give the right direction, 26% give the direction with the error of $\pm 8°$.

3 Path Generation

Although we have designed the image processing system with angle resolution of 8 degree so far, the accuracy is not enough to control the direction of the robot. Inaccuracy of the direction should be absorbed in the path generation system.

The path generation system generates and evaluates a path from a current position to a target position, where the target position is given by the strategy system[4]. Moreover, since our robot has only 2 wheels, there is a constraint to control the direction.

Taking account of these conditions, we have designed a path generation system.

3.1 System Configuration

A path generation system consists of a path generator, a path evaluator, an auxiliary target point generator, a simple predictor, and a velocity generator, as shown in figure 8. This system works as follows,

1. The path generation system receives the target position, the robot direction and the action (stop, shoot etc) at the target position from a strategy system.

2. The path generator generates a working path by using a curvature control variable and sends it to the path evaluator. The path evaluator evaluates the working path from a required time to arrive at the target position and from a possibility of violating RoboCup soccer rules.

3. Determine an advancing direction of a robot based on an evaluation result.

4. If there is an obstacle on the path or if robot direction at the target does not satisfy the given direction, the auxilialy target point is added for avoiding collision or for satisfing the direction at the target. Then, a new path is calculated again.

5. The new path is evaluated again.

6. Iterate step 3 - 5 by modifing the curvature control variable, and get an optimal path.

7. The velocity generator generates the velocities of wheels which move on the optimal path.

8. The simple predictor corrects the velocities of wheels to compensate the delay of processing time of image processing system. The corrected velocities are sent to the robot through a radio system.

3.2 Path generator

Since our robot has two wheels, the degree of control freedom is two and the control of direction is limited. Considering this and the fact that the target position where each robot should go changes from hour to hour, it is realistic to genarate a path which the robot approaches the target position by asymptotically changing his direction. We call this a **direction converging path generation**. Figure 9 shows the paths generated by this method. In the figure, paths are superimposed on the field image. Each double circle with a number is a current position of our robot and the end point of each path is a target position. The target position with triple circle is a ball. The dotted circles are opponent robots. If opponent robots stand on the generated path, subtargets are put near the opponents to avoid collision. In the case, the robot goes to the subtargets at first and then goes to the target. The robots number 2 and 3 have a subtarget and the robots number 0 and 1 do not. The goal keeper (number 4) does not move in this figure. Note that the path is generated only to determine the velocity for next Δt time step[2] . The path is newly generated every Δt time step.

Our path generation algorithm is given as follows.

step1 Let $p(=(x,y)),(v_l,v_r),\mathbf{u}$ be position, velocity (of left and right wheels) and forward direction vector of a robot, respectively. Let t be the current time. Calculate the curvature κ and robot velocity $v = \frac{v_l+v_r}{2}$. See literature [5] for detail.

step2 Get a goal position $p' = (ox, oy)$. This is given by the strategy algorithm.

step3 Let $\overrightarrow{pp'}$ be a vector directed from the current position to the goal position and let θ be an angle between vectors $\overrightarrow{pp'}$ and $\mathbf{u}$.

step4 Give a curvature at the time $t+\Delta t$ by $\kappa_{new} = \theta \times n_a/R_A$, where R_A is a constant and n_a is a variable depending on subgoal(s). (This equation generates a path which has a large curvature at first and a small as approaching to the goal. See fig. 9.)

step5 Putting $dr = 1/\kappa - 1/\kappa_{new}$, calculate a velocity variation of robot $|dv| = |S/dr|$, where S is a constant. Let $v_m(\kappa)$ be a maximum robot speed when a curvature κ is given. Give new velocity by $v_{new} = v + dv$ if $v_{new} < v_m(\kappa_{new})$, otherwise by $v_{new} = v - dv$. Then, calculate a new position at the time $t+\Delta t$.

[2] Image processing speed determines the Δt. In our system, $\Delta t = 33msec$.

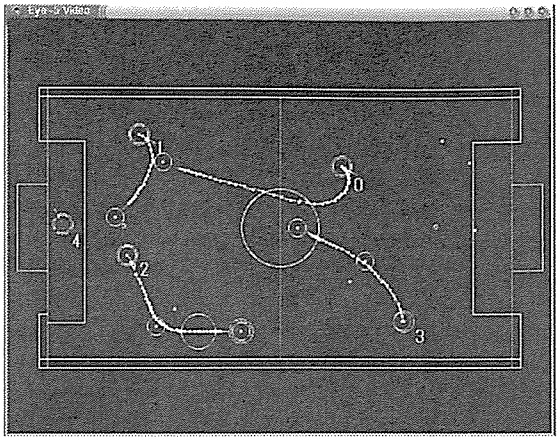


Figure 9: An exapmle of path generation

step6 Calculate repeatedly the steps from *step1* to *step5* and check whether the path reaches the given goal or not. (Fig. 9 shows the result of this calculation.) If the path reaches the goal, it is OK. If not (if over M times repeated), recalculate these steps by changing the constant R_A until the path reaches the goal. This computation gives the robot velocity of next Δt time period.

4 Concluding remarks

In this paper, we proposed a fast and robust image processing method which is coped with the variance of color parameters in the field and a new labeling algorithm, "diagonal alternate spread labeling algorithm". It was clarified experimentally that these method and algorithm are very effective and robust to extract moving objects up to 240cm/sec without any condition that it is an interlace format image or not.

We also show the path generation algorithm in which the robot approaches the goal by changing its direction convergently. By using these two algorithms, real time system of 1/30sec is realized.

Although our system installed these algorithms works well in real time, but there are remaining issues. As is improved to be robust for spatial variance, it is necessary to examine and improve the image processing algorithm to be more robust for time variance of lighting condition of the game field. And, it is also important to refine the angular precision of each robot to decrease calculation error. These are our future subjects.

Acknowledgement

This paper was partially supported by Grant-in-Aid for General Scientific Research (C) of Ministry of Education, Culture, Sports, Science and Technology, and the Tatematsu Foundation and AI Research Promotion Foundation.

References

[1] M. Veloso et al eds. "Lecture Notes in Artificial Intelligence 1856, RoboCup-99: Robot Soccer World Cup III", Springer, 2000

[2] P. Stone et al eds. "Lecture Note in Artificial Intelligence 2019, RoboCup 2000: Robot Soccer World Cup IV", Springer, 2001

[3] J. Bruce, T. Balch, M. Veloso "Fast and Inexpensive Color Image Segmentation for Interactive Robots", Proc. 2000 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 2061 - 2066, 2000

[4] Y.Nagasaka, K.Murakami, T.Naruse, T. Takahashi and Y.Mori: "Potential Field Approach to Short Team Action Planning in RoboCup F180 League", in P. Stone et al eds., Lecture Note in Artificial Intelligence 2019, RoboCup 2000: Robot Soccer World Cup IV, pp.345-350, Springer (2001).

[5] R. D'Andrea and J. Lee "Cornell Big Red Small-Size-League Winner", AI magazine, Vol. 21, No. 3, pp. 41 - 44, 2000

Behavior Acquisition based on Multi-Module Learning System in Multi-Agent Environment

Yasutake Takahashi, Kazuhiro Edazawa, and Minoru Asada
Emergent Robotics Area, Dept. of Adaptive Machine Systems,
Graduate School of Engineering, Osaka University
{yasutake,eda,asada}@er.ams.eng.osaka-u.ac.jp

Abstract

The conventional reinforcement learning approaches are difficult to handle the policy alternation of the opponents because it may cause dynamic changes of state transition probabilities of which stability is necessary for the learning to converge. This paper presents a method of multi-module reinforcement learning in a multiagent environment, by which the learning agent can adapt itself to the policy changes of the opponents. We show a preliminary result of a simple soccer situation in the context of RoboCup.

1 Introduction

There have been an increasing number of approaches to robot behavior acquisition based on reinforcement learning methods. The conventional approaches need an assumption that the environment is almost fixed or changing slowly so that the learning agent can regard the state transition probabilities are consistent during its learning. Therefore, it seems difficult to apply the reinforcement learning method to a multiagent system because a policy alteration of the other agents may occur, which dynamically changes the state transition probabilities from the viewpoint of the learning agent. RoboCup provides such a typical one, that is, a highly dynamic, hostile environment, in which an agent has to obtain purposive behaviors.

There are a number of work on reinforcement learning system in a multiagent environment. Asada et al. [1] proposed a method which estimates the state vectors representing the relationship between the learner's behavior and those of other agents in the environment using a technique from system identification, then reinforcement learning based on the estimated state vectors is applied to obtain the optimal behavior policy. However, this method requires re-learning or adjustment of learning agent's policy whenever the other agents change their policies, even if they switch their policies back which the learning agent has already adjusted before. This problem happens because one learning module can maintain only one policy.

A multiple learning module approach would provide one solution for this problem. If we can assign multiple learning modules to different situations in which the each module can regard the state transition probabilities are consistent, then the system would provide reasonable performance. There are a number of work on the multi-learning module systems.

Singh [2, 3] has proposed compositional Q-learning in which an agent learns multiple sequential decision tasks with multi learning modules. Each module learns its own elemental task while the system has a gating module for the sequential task, and this module learns to select one of the elemental task modules. Takahashi and Asada [4] proposed a method by which a hierarchical structure for behavior learning is self-organized. The modules in the lower networks are organized as experts to move to different categories of sensor value regions and learn lower level behaviors using motor commands. In the meantime, the modules in the higher networks are organized as experts which learn higher level behavior using lower modules. Each module assigns its own goal state by itself. However, there are no such measure to identify the situation that the agent can change modules correspond-

ing to the current situation.

Sutton [5] has proposed DYNA-architectures which integrate world model learning and execution-time planning. Singh [6] has proposed a method of learning a hierarchy of models of the DYNA-architectures. The world model is not for the identification of the situations, but only for improving the scalability of reinforcement learning algorithms.

Doya et al. [7] have proposed MOdular Selection and Identification for Control (MOSAIC), which is a modular reinforcement learning architecture for non-linear, non-stationary control tasks. The basic idea is to decompose a complex task into multiple domains in space and time based on the predictability of the environmental dynamics. Each module has a state prediction model and a reinforcement learning controller. The models have limited capabilities of state prediction as linear predictors, therefore the multiple prediction models are required for the non-linear task. A domain is specified as a region in which one linear predictor can estimate sensor outputs based on its own prediction capability. The responsibility signal is defined by a function of the prediction errors, and the signals of the modules define the outputs of the reinforcement learning controllers. Haruno et al. [8, 9] have proposed another implementation of MOSAIC based on multiple modules of forward and inverse models.

In this paper, we propose a method by which multiple modules are assigned to different situations and learn purposive behaviors for the specified situations as results of the other agent's behaviors. We show a preliminary result of a simple soccer situation in the context of RoboCup.

2 A Basic Idea and An Assumption

The basic idea is that the learning agent could assign one reinforcement learning module to each situation if it can distinguish a number of situations in which the state transition probabilities are consistent. We introduce a multiple learning module approach to realize this idea. A module consists of learning component which models the world and an execution-time planning one. The whole system will follow these procedure simultaneously.

- find a model which represents the best estimation among the modules,
- update the model, and
- calculate action values to accomplish a given task based on DP.

As a preliminary experiment, we prepare a case of ball chasing behavior with collision avoidance in the context of RoboCup. The problem here is to find the model which can most accurately describe the opponent's behavior from the view point of the learning agent. It may take a time to distinguish the situation, then, we put an assumption.

- The policy of the opponent might change match by match but is fixed during one match.

3 A Multi-Module Learning System

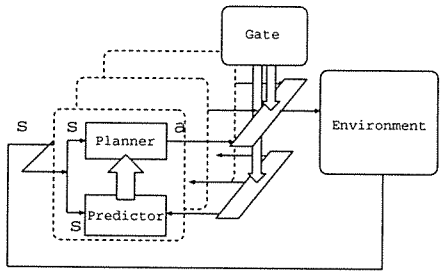


Figure 1: A multi-module learning system

Fig.1 shows a basic architecture of the proposed system, that is, a multi-module reinforcement learning one. Each module has a forward model (predictor) which represents the state transition model, and a behavior learner (policy planner) which estimates the state-action value function based on the forward model in the reinforcement learning manner. This idea of combination of a forward model and a reinforcement learning system is similar to the H-DYNA architecture [6] or MOSAIC [7, 8, 9]. In other words, we extend such architectures to an application of behavior acquisition in the multi-agent environment.

The system selects one module which has the best estimation of the state transition sequence by activating a gate signal corresponding to a module and by deactivating the goal signals of other modules, and the selected module sends action commands based on its policy.

3.1 Predictor

In this experiment, the agent recognizes a ball, a goal, and the opponent in the environment. The state space of the planner consists of features of all objects in order to calculate state value (discounted sum of the re-

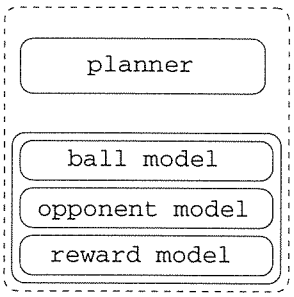


Figure 2: An architecture of a module

ward received over time) for each state and action pair. However, it is impractical to maintain a full size state transition model for real robot applications because the size of state-action space becomes easily huge and it is really rare to experience all state transitions within the reasonable learning time.

In general, the motion of the ball depends on the goal and the opponent because there are interactions between the ball, the goal, and the opponent. However, the proportion of the interaction time is much shorter than that of non-interaction time. Therefore, we assume that the ball motion is independent from the goal and the opponent. Further, we assume that the opponent motion from the viewpoint of the agent seems independent from the ball and the goal positions and to depend on only the learning agent's behavior even if the opponent's decision may depend on the ball and/or the goal positions. If the system has maintain the forward models of the ball, the goal, and the opponent separately, the each models can be much compact and it is easy to experience almost state transition within reasonable learning time.

Fig.2 shows an architecture of one module in our system. As mentioned above, the module has three forward models for the ball, the goal, and the opponents. We estimate the state transition probability $\hat{\mathcal{P}}^a_{ss'}$ for the triplet of state s, action a, and next state s' using the following equation:

$$\hat{\mathcal{P}}^a_{ss'} = \hat{\mathcal{P}}^a_{{}^b s {}^b s'} \cdot \hat{\mathcal{P}}^a_{{}^g s {}^g s'} \cdot \hat{\mathcal{P}}^a_{{}^o s {}^o s'} , \qquad (1)$$

where the state $s \in S$ is a combination of three states in the ball state space ${}^b s \in {}^b S$, the goal state space ${}^g s \in {}^g S$, and the opponent state space ${}^o s \in {}^o S$. The system has not only the state transition model but also a reward model $\hat{\mathcal{R}}^a_{ss'}$.

We simply store all experiences (state-action-next state sequences) to estimate these models. According to the assumption mentioned in **2**, we share the state transition models of the ball and the goal and the reward model among the modules, and each module has its own opponent model. This leads further compact model representation.

3.2 Planner

Now we have the estimated state transition probabilities $\hat{\mathcal{P}}^a_{ss'}$ and the expected rewards $\hat{\mathcal{R}}^a_{ss'}$, then, an approximated state-action value function $Q(s,a)$ for a state action pair s and a is given by

$$Q(s,a) = \sum_{s'} \hat{\mathcal{P}}^a_{ss'} \left[\hat{\mathcal{R}}^a_{ss'} + \gamma \max_{a'} Q(s',a') \right] , \qquad (2)$$

where $\hat{\mathcal{P}}^a_{ss'}$ and $\hat{\mathcal{R}}^a_{ss'}$ are the state-transition probabilities and expected rewards, respectively, and the γ is the discount rate.

3.3 Gating Signals

The basic idea of gating signals is similar to Tani and Nolfi's work [10] and the MOSAIC architecture [7, 8, 9]. The gating signal of the module become larger if the module does better state transition prediction during a certain period, else smaller. We assume that the module which can does best state transition prediction has the best policy against the current situation because the planner of the module is based on the model which describes the situation best. In our proposed architecture, the gating signal is used for following purposes:

1. gating the learning of prediction models
2. gating the action outputs from modules

We calculate the gating signals g_i of the module i as follows:

$$g_i = \frac{e^{\lambda p_i}}{\sum_j e^{\lambda p_j}}$$

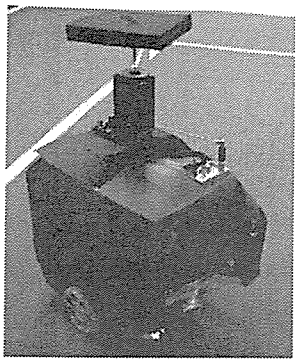
where p_i is the occurrence probability of the state transition from the previous state to the current one according to the model i.

4 Experiments

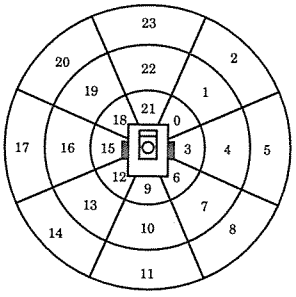
We have studied preliminary experiments so far. The task of the learning agent is to catch the ball while it avoids the collision with the opponent.

4.1 Setting

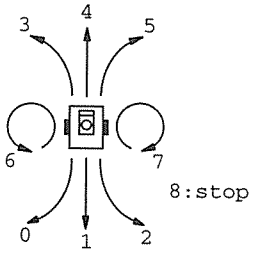
Fig.3(a) shows a picture in which the mobile robot we have designed and built. A simple color image processing (Hitachi IP5000) is applied to detect the ball area in the



(a) Robot



(b) State space



(c) Action space

Figure 3: experiment

image in real-time (every 33ms). The state space is constructed in terms of the centroid of the ball and the opponent on the image (Fig.3(b)). The driving mechanism is PWS (Power Wheeled System), and the action space is constructed in terms of two torque values to be sent to two motors corresponding to two wheels (Fig.3(c)). These parameters of the robot system are unknown to the robot, and it tries to estimate the mapping from sensory information to appropriate motor commands by the method.

The opponent has a number of behaviors such as "stop", "move left", and "move right", and switch them randomly after a fix period.

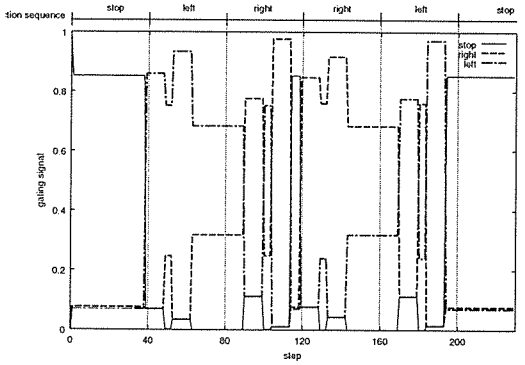


Figure 4: An example sequence of the gating signals

4.2 Result

First of all, we let the learning agent have models to those behaviors of the opponents. The learning agent behaves randomly while it gathers the data of the ball and the opponent image positions and builds up the models of them. Next, the learning agent tries to estimate the opponents behavior through the sequence of the observation. Fig. 4 shows an example sequence of the opponent's behavior estimation while the learning agent is stationary and the opponent behaves randomly after a fixed period.

5 Conclusion

In this paper, we proposed a method by which multiple modules are assigned to different situations and learn purposive behaviors for the specified situations as results of the other agent's behaviors. We have shown a preliminary result of a simple soccer situation in the context of RoboCup.

Currently, we are struggling with implementation the proposed method into one to one match, two to two, and so on. We hope we can show more interesting results at the symposium.

参考文献

[1] M. Asada, E. Uchibe, and K. Hosoda. Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development. *Artificial Intelligence*, 110:275–292, 1999.

[2] Satinder Pal Singh. Transfer of learning by composing solutions of elemental sequential tasks. *Machine Learning*, 8:323–339, 1992.

[3] Satinder P. Singh. The effeicient learnig of multiple task sequences. In *Neural Information Processing Systems 4*, pages 251–258, 1992.

[4] Y. Takahashi and M. Asada. Vision-guided behavior acquisition of a mobile robot by multi-layered reinforcement learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 395–402, 2000.

[5] Richard S. Sutton. Integrated modeling and control based on reinforcement learning and dynamic programming. *Advances in Neural Information Processing Systems 3*, pages 471–478, 1991.

[6] Satinder P. Singh. Reinforcement learning with a hierarchy of abstract models. In *National Conference on Artificial Intelligence*, pages 202–207, 1992.

[7] Kenji Doya, Kazuyuki Samejima, Ken ichi Katagiri, and Mitsuo Kawato. Multiple model-based reinforcement learning. Technical report, Kawato Dynamic Brain Project Technical Report, KDB-TR-08, Japan Science and Technology Corporation, June 2000.

[8] Masahiko Haruno, Daniel M. Wolpert, and Mitsuo Kawato. Multiple paired forward-inverse models for human motor learning and control. *Advances in Neural Information Processing Systems*, 11:31–37, 1999. MIT Press, Cambridge, Massachusetts.

[9] Masahiko Haruno, Daniel M. Wolpert, and Mitsuo Kawato. Mosaic model for sensorimotor learning and control. *Neural Computation*, 13:2201–2220, 2001.

[10] Jun Tani and Stefano Nolfi. Self-organization of modules and their hierarchy in robot learning problems: A dynamical systems approach. Technical report, Technical Report: SCSL-TR-97-008, 1997.

A Comparison of the Self-Localization methods and a Method for Cooperative Behaviors Using the Environment Map

自己位置同定法の比較と環境マップを用いた協調動作の方法について

Hidetomo Suzuki, Hiroshi Nakano, Masahito Osanai,
Shintaro Yasui, Ichiro Watanabe, Jiro Kashio
鈴木秀智, 中野博史, 長内真人, 安井慎太郎, 渡邉一郎, 樫尾次郎
Mie University
三重大学
{suzuki,nakano,osanai,yasui,wata,kashio}@kashio.info.mie-u.ac.jp

Abstract

A robot must know its surroundings when it carries out the given tasks with cooperation. In this paper, we compares three kinds of self-localization methods, proposes a method suitable for the robot with only one monocular camera system based on the comparison, and shows how to make the environment map. The environment map is constructed using local visual information of each robot, describes the situation of the soccer field and is used to accomplish the cooperative behaviors. With the environment map, we can realize the cooperative behavior, that is, each of our robots stays at the favorable position in strategy, instead of running after the ball and being bunching up together.

1 Introduction

It is indispensable that each of robots knows its surroundings such as the position of the ball and goals when the robots carry out cooperative tasks. It is rather hard for each of robots to get and keep the information on its surroundings separately if the robot has only one video camera with the limited viewing angle. On the other hand, it is necessary that a robot keeps its own position to estimate the arrangement of objects and share it among its teammates. There are many studies on self-localization and sharing information. For example, CS Freiburg Robotic Soccer Team [1] uses the method based on a laser range finder (LRF) for self-localization. The method acquires the distance to objects using LRF, makes the line segments, matches them against an model of the soccer field. Recently, many of the middle size RoboCup teams introduced omni-directional camera systems, and are utilizing the system to recognize the objects in the soccer field such as lines and goals [2] - [8].

We looked for the suitable self-localization method for our robotic soccer players which has only a monocular camera with the limited viewing angle when we designed and assembled them, because we needed the method to make the accurate environment map.

In the next section, we give an overview of the architecture of our robotic soccer players. Sect. 3 focuses on the comparison of three self-localization methods and our method constructed based on the comparison. In Sect. 4, we describe the method of making the environment map by integrating the visual information of each robot. In Sect. 5 we conclude.

2 Architecture of Our Robots

The components of our robots are the commonly used products, and the architecture of the robots is similar to that of the other teams. But, on designing our robots, we adopted the following concepts.

- They should have the powerful processing units sufficient for image processing, a sense-plan-do loop and a control of a mobile base,
- All the tasks of the system should act like objects working independently so that we can easily develop and improve the system.
- All the tasks of the whole multi-robots system should communicate with each other in a seamless way.

For the first prerequisite, we composed our robot with an image processing board for getting visual information,

a mobile base unit (platform) including a controller for controlling its movement, and an industrial single board computer for other tasks such as recognition of its surroundings, planning its behaviors and the communication with other robots. For the latter two prerequisites, we adopted the PVM (Parallel Virtual machine) library [9] which constructs a virtual parallel computer on various computers connected to a LAN.

Each of our robot is composed of a control unit and a mobile base unit (a robot platform) (See Fig. 1). The control unit includes an image processing part and a system control part. The former part is equipped with an image processing board and executes image processing to detect and localize objects in the soccer field. The latter part is equipped with an industrial single board computer and works for recognizing the circumstances around the robot, planning its behaviors, transmitting commands and communicating with other robots. We use TRIPTERS-mini manufactured by JSD as the mobile base unit. It is a mobile robot with one steering wheel, two driving wheels, a control unit and 8 ultrasonic sonar (US) sensors. It is equipped with a 16 bit CPU (NEC, V25) for controlling wheels and sensor units. The CPU communicates with the control unit via RS-232C interface. It acts as the controller of the primitive actions ordered by the control unit, and sends US sensor data when the control unit requests them.

Our robot mounts the sensory system, including a video camera and US sensors. The video camera (Sony, EVI-D30) has a motorized pan-tilt-zoom unit and has the 60 degree horizontal field of view. It sends NTSC video signal to the image processing board (Hitachi, IP-5005BD) which converts the video signal to 256 by 220 color digital image, and extracts and measures objects in every 66 msec. 8 US sensors are used for detection of the neighboring objects like the ball and the wall, and measurement of the distance to them. US sensor data are mainly used for collision avoidance.

The following tasks work with the system control part in a robot (Fig. 2). The task MAP runs in one of our robots.

- COG : recognizes the environment around a robot, and communicates with other robots,
- PLAN : makes behavior plans, and sends commands to other tasks,
- TRANS : controls mobile base unit,
- CAMERA : controls EVI-D30,

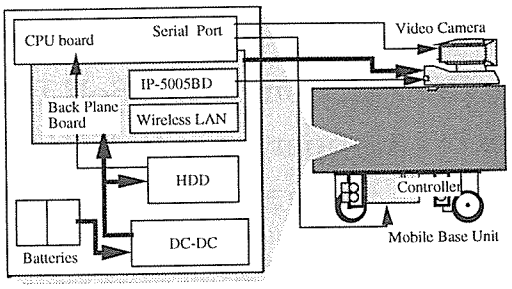


Figure 1: Hardware Architecture of Our Robotic Soccer Player

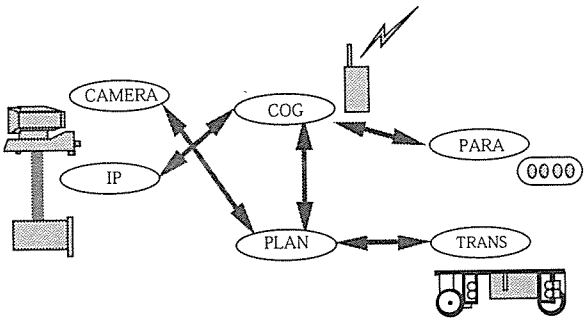


Figure 2: Configuration of Tasks

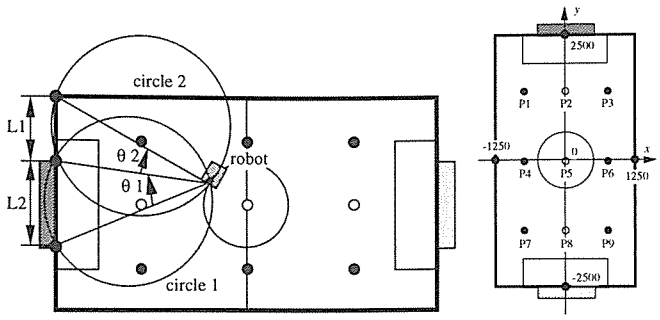
- IP : controls IP-5005BD and manages processing results,
- PARA : controls PC's parallel ports,
- MAP : makes the environment map.

We use Linux as its operating system, and PVM for managing multi-processes. The average time to execute one Sense-Plan-Do loop is 66.7 msec.

3 Self-Localization

Our robot records the initial position given at the starting point, and updates by odometry, that is, using the values obtained from the rotary encoders attached to the driving wheels of the mobile base unit. It determines the initial position using one of the following methods in accordance with its condition. The first one is the method that utilizes three landmarks known their positions previously like goals and corners, and the second one is the method that utilizes the range data of US sensors and local visual information.

Odometry is reliable in the RoboCup soccer field unless the robot is disturbed. But, it is possible that the robot suffers from disturbances which makes its odometry inaccurate. In such a case, the robot examines the accuracy of its estimated position by using the position obtained by the self-localization. If the accuracy of the



(a) Calculation (b) Positions

Figure 3: Self-localization (Method 1).

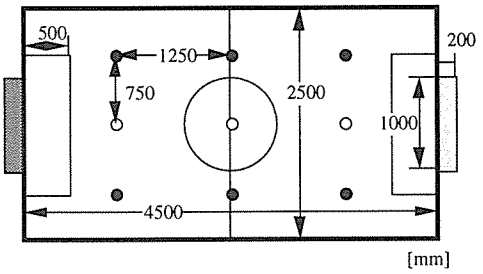


Figure 4: The Field Used in the Experiment

estimated position is no longer reliable the robot uses a self-localization method, and corrects it.

We evaluated three typical self-localization methods in order to develop the suitable method for our robots. We did not use the methods based on matching line segments because of their processing time, nor used the methods utilizing the apparent size in an image frame because they need the actual size of the objects.

The followings are the results of the evaluations.

3.1 Method 1

This method is a popular one to estimate the position using landmarks (Fig. 3 (a)). For example, Nakamura et al. [8] uses this kind of method.

1. A robot, whose position is to be estimated, selects three landmarks which are not colinear, and measures the orientation from the robot.
2. It chooses two sets of two points out of the landmarks, and makes two circumscribed circles. Each of the circles includes two points of the set and the robot.
3. One of the intersections of two circles is the position of the robot.

We evaluated the precision of this method. We used the soccer field which is half the size of the field of the RoboCup middle size league. See Fig. 4.

In the first experiment, we chose both of the blue side goal posts and the left corner post of the blue side as the landmarks. The result of the experiment is shown in Table 1. In the second experiment, we chose the right post of the blue goal, the right corner post of the blue side and the left corner post of the yellow side as the landmarks, so that the including angles are large. The result of the experiment is shown in Table 2. P_i, the position of the robot, is depicted in Fig. 3 (b). The row Δ x shows the average estimation errors in x coordinate, and the row Δ y shows those in y coordinate.

Those results shows that the estimation error grows larger as the robot leaves the landmarks more in the distance. This is because the included angle made by two lines each of which connects the robot and one of the landmarks becomes acute as the robot leaves the landmarks, the error of its position calculated using the coordinates in an image gets worse, and it becomes difficult to distinguish the coordinates observed at different positions. Thus the robot must search for the landmarks as near as possible when it utilizes this method.

In the second experiment, the errors tend to be small at several positions than those of the first one. But the robot needs at least two image frames to know the directions of landmarks because it is difficult to put the landmarks needed for the localization in a single image frame. Though the robot can use two sets of the landmarks with large included angle by moving the camera and the body, the accuracy of the estimated position gets worse. We must choose the appropriate method for selecting landmarks that tades off the aquisition time and the accuracy.

Nakamura et al. [8] uses this kind of method in their autonomous robots with an omnidirectional camera. They reported the experimental result that the average estimation errors were 284 mm in x direction, 292 mm in y direction and 15 ° in orientation. We must halve those values to compare with our results becaus of the size of our field. The average errors of their report are better than our results. It is because a robot with an omnidirectional camera can select the landmarks appropriate for the localization in a single image frame. But we think that our method which searches the landmarks actively is a realistic solution for the robot with a monocular camera provided that the acquisition time does not degrades its behaviors seriously.

Table 1: The result of the first experiment of method 1. The estimation errors are shown in [mm].

	P1	P2	P3	P4	P5	P6	P7	P8	P9
Δ x	147	33	75	508	2	252	761	332	648
Δ y	33	12	18	29	2	44	17	12	188

Table 2: The result of the second experiment of method 1. The landmarks were chosen in order to make the angles as large as possible. The estimation errors are shown in [mm].

	P1	P2	P3	P4	P5	P6	P7	P8	P9
Δ x	8	77	250	28	3	159	4	34	82
Δ y	19	143	27	3	106	124	0	137	563

3.2 Method 2

This method uses the distances and orientations of the two landmarks. This method makes two circles using them, gets the intersection of the circles, and determines the position. This method is more favorable than other methods in the point that two landmarks tend to be observed in a sigle image frame, and the robot does not need to move its camera or itself to find out the landmarks. This method is similar to Jamzad et al. [11] and Bandlow et al. [12].

The method consists of the following steps. The symbols are shown in Fig. 5.

1. The coordinates of two landmarks, A and B, are extracted from an image frame.
2. The distances d(OA') and d(OB') are estimated, where ' means the projection of a point to the principal axis of the camera. O denotes the position of the robot.
   (a) a_y and b_y, the y coordinates of those landmarks, are taken from the image frame.
   (b) d(OA') and d(OB') are calculated using a_y and b_y, respectively. A predefined function which converts a y coordinate to the distance is used.
   (c) θ_a and θ_b are calculated using a_x and b_x, respectively, where θ_a is the angle made by the principal axis of the camera and the line segment OA.

$$\theta_a = \frac{30}{180}|(a_x - 128)|, \qquad \theta_b = \frac{30}{180}|(b_x - 128)| \tag{1}$$

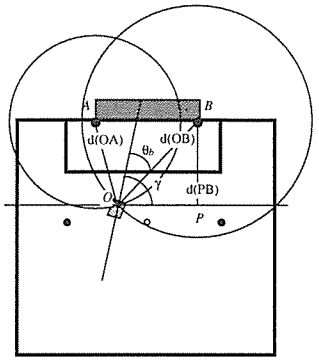


Figure 5: Self-Localization (Method 2)

Table 3: The result of of Method 2. The estimation errors of Δ x and Δ y are shown in [mm], and those of $\Delta\theta$ is shown in [°].

	P1	P2	P3	P4	P5	P6
Δ x	193	32	168	293	122	241
Δ y	195	83	274	104	93	20
$\Delta\theta$	1	4	1	2	5	2

   (d) The distances d(OA) and d(OB) are calculated using the following equation.

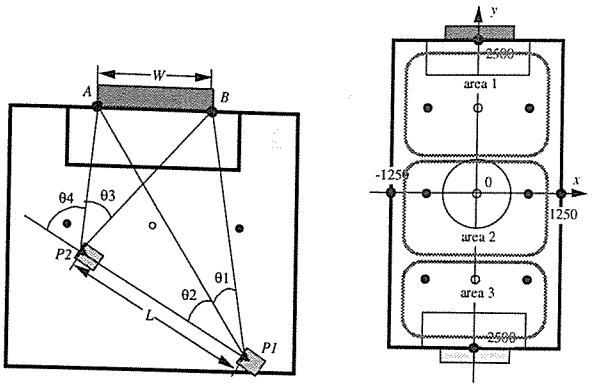
$$\mathrm{d}(X) = \frac{\mathrm{d}(X')}{\cos\theta} \tag{2}$$

   where, X is OA or OB.

3. Two circles are defined using the coordinates of the goal posts and their distance. The radius of one of them is d(OA), and its center is at A. The radius of the another one is d(OB), and its center is at B. The coordinates of the robot is detemined as the intersection of those circles. Though each of the circles intersects at two points, it is possible to choose the true position because the false one lies outside the field.
4. The orientation γ of the robot is calculated using θ_b and d(PB), where d(PB) is the distance from B to P which is the projected point of B to the line which is parallel to the goal line and passes through O.

$$\gamma = \theta_b + \sin^{-1}\left(\frac{PB}{OB}\right) \tag{3}$$

Table 3 shows the experimental result of Method 2. The average errors are worse than those of Method 1. The reason for the errors is that the resolution is insufficient for determining the distance like d(OA).



(a) Caluculation (b) Areas

Figure 6: Self-Localization (Method 3)

Table 4: The result of of Method 3. The estimation errors of Δ x and Δ y are shown in [mm].

	Area 1	Area 2	Area 3
Δ x	109	277	747
Δ y	150	212	318

3.3 Method 3

This method uses the directional information observed at two points. A robot obtains the orientation angles to two landmarks at each point. The method requires the accurate movement because the distance of movement L is used to the localization. We referred the formulation of Egami et al. [10].

This method consists of the following steps. The symbols are shown in Fig. 6 (a).

1. The orientations to both of goal posts, θ_1 and θ_2, are calculated at P_1.
2. The robot moves to P_2 apart from P_1 by L mm.
3. The orientations to both of goal posts, θ_3 and θ_4, are calculated at P_2.
4. The orientations, the distance between the goal post W and L are used to calculate the position.

Table 4 shows the experimental result of Method 3. The areas are shown in Fig. 6 (b). The distance L was changed corresponding to the area to keep the measured angle large enough. L was 1000 mm in the area 1 and 2, and 1500 mm in the area 3. The average errors are large because the angles observed do not show the significant difference. To reduce the error, it might be necessary to enlarge L or use more observing points.

Based on the results of those experiments, we adopted the method that usually uses the position obtained by odometry, and uses the estimated position obtained by using the landmarks searched actively when the odometry becomes unreliable. By this method, we could reduce the estimation error of self-localization.

4 Environment Map

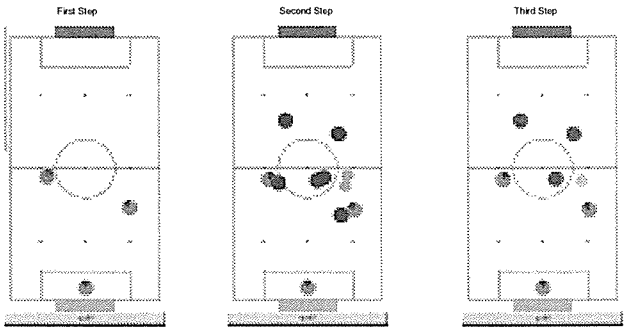
The environment map describes the field and the positions of objects such as a ball, goals and robots, and is updated almost periodically. It is made by integrating the visual information of teammates, and then dispatched to each of them. It is similar to the *Global Map* of Gutmann et al. [1]. With it, our robot can know the situation outside its view, and the problem of occlusion is reduced very much. Moreover, robots become more intelligent so that each of them can stay at the favorable position in strategy, instead of following the ball and being bunching up together.

4.1 Make of Environment Map

The algorithm of making the environment map consists of the following steps (Fig. 7).

1. A robot which acts as a map maker plots the self-localized positions obtained from its teammates on the map (Fig. 7 (a)).
2. The positions of the objects like the ball and opponents are added to the map (Fig. 7 (b)). The positions of objects obtained from teammates must be translated to the world coordinate frame of the map, because the positions are epressed in the local coordinate frame.
3. The position of the ball is determined using the gravity center of the multiple points of the ball.
4. The positions of opponents are determined using a nearest neighbor method. The resultant clusters are ranked according to the number of points belonging to the cluster, and the candidates of the opponents selected based on the rank (Fig. 7 (c)).

Fig. 7 is an example of this method. There are two balls in Fig. 7 (b), which appear due to the estimation errors of two teammates. An object nearby one of the teammates is the virtual opponent made by estimation error, too. Those false objects are eliminated in Fig. 7 (c).



(a) (b) (c)

Figure 7: Steps of Making a Map. (a) Teammates are plotted. (b) The ball and opponents are plotted using the local visual information. (c) Those are determined by eliminating false objects.

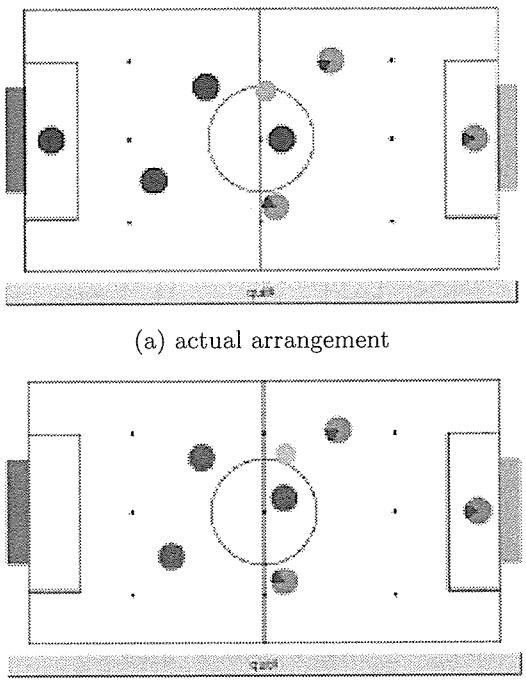
Table 5: Estimation error of the environment map. Values are shown in [mm].

	ball	opponents
Δ x	53	69
Δ y	65	88

4.2 Experiment of Making the Environment Map

In the experiment, several arrangements of objects are used to make the environment map. Fig. 8 (a) is an example of the arrangements, and Fig. 8 (b) is the map obtained using the proposed method. The robot at the left side is not shown in the map because it can be seen by none of the teammate robots. The rest of the robots are plotted on the map. The estimation errors of the objects are good enough to recognize the situation of the field (Table. 5). The average period which includes requesting the information of self-localization, calculation and transmitting the map was 180 msec, and its standard deviation was 29 msec. The average period of one cycle of Sense-Plan-Do loop is about 100 msec, so that the environment map is updated at the rate of once per two cycles of the loop. This implies that our method has the accuracy enough for planning cooperative behaviors.

The positions of not only teammates but also opponents could be known by integrating the visual information of each robot, and moreover the problem of occlusion could be reduced. However, when an object disappears from the map, the interpolation between image frames used in our method did not give sufficient estimation. We must improve it by introducing another method like a method based on Kalman filtering (Gutmann et al. [1]), or a method using probability distribution.



(a) actual arrangement

(b) estimated arrangement

Figure 8: Result of an Experiment.

It is difficult to make the environment map at the specified time because the imaging time of the image from each robot is not same. Gutmann et al. [1] adopted a method which synchronizes it by using the time code, but still we think it is very difficult to synchronize at the deviation within several mili seconds.

5 Conclusions

A robot must know its surroundings when it carries out the given tasks with cooperation. In this paper, we compared three kinds of self-localization methods, proposed a method suitable for the robot only with a monocular camera system, and showed how to make the environment map. The most part of these methods was developed referring to the methods which had already been reported. We adopted a self-localization method that chose landmarks which were the most suitable for knowing the own position actively, and estimated the position.

The self-localization and the making map we proposed are reasonable methods, though these takes much time to determine the position because those uses the monocular camera with the restricted viewing area. Though the precision of estimated position could be improved by choosing the suitable landmarks, the error arose in

the position apart from the landmarks, and was equal to the size of the robots. To solve this problem, we think we must enlarge the size of the image to increase the resolution and reduce the calibration error of the camera.

The latency due to the acquisition of the local visual information in each robot and the making the environment map influences the response speed of a robot. But, it can be thought that our method is practically enough for playing in the middle-size league, because the error due to the synchoronous deviation and the latency due to image processing are smaller than the error of the position estimation by image processing. Of cource, the reduction of those errors should be solved as a future subject.

At present, we realised a role sharing using the environment map, that is, each of our robots can stay at the favorable position in strategy, instead of following the ball and being bunching up together.

We are planning to realize the effective pass work, introduce the learning algorithm which obtains the advanced behaviors, and so on, as the future work.

参考文献

[1] Gutmann, J.-S, Hatzack, W., Herrmann, I., Nebel, B., Rittinger., F., Topor, A., Weigel T., Welsch, B.: The CS Freiburg Robotic Soccer Team: Reliable Self-Localization, Multirobot Sensor Integration, and Basic Soccer Skills. Lecture Notes in Artificial Intelligence, Vol. 1604. Springer-Verlag, Berlin Heidelberg New York (1998) 93–108

[2] Price, A.R., Jones, T.: An innovative Approach to Vision, Localization and Orientation Using Omnidiectional Radial Signature Analysis. Lecture Notes in Artificial Intelligence, Vol. 1604. Springer-Verlag, Berlin Heidelberg New York (1998) 299–315

[3] Suzuki, S., Kato, T., Ishizuka, H., Takahashi, Y., Uchibe, E., Asada, M.: An Application of Vision-Based Learning in RoboCup for a Real Robot with an Omnidirectional Vision System and the Team Description of Osaka University "Trackies". Lecture Notes in Artificial Intelligence, Vol. 1604. Springer-Verlag, Berlin Heidelberg New York (1998) 316–325

[4] Cassinis, R., Rizzi, A.: Design Issues for a RoboCup Goalkeeper. Lecture Notes in Artificial Intelligence, Vol. 1856. Springer-Verlag, Berlin Heidelberg New York (1999) 254–262

[5] Plagge, M., Günther, R,. Ihlenburg, J., Jung, D., Zell, A.: The Attempto RoboCup Robot Team. Lecture Notes in Artificial Intelligence, Vol. 1856. Springer-Verlag, Berlin Heidelberg New York (1999) 424–433

[6] Marques, C.F., Lima, P.U.: A Localization Method for a Soccer Robot Using a Vision-Based Omni-Directional Sensor. Lecture Notes in Artificial Intelligence, Vol. 2019. Springer-Verlag, Berlin Heidelberg New York (2000) 96–107

[7] Marchese, F.M., Sorrenti, D.G.: Omni-directional Vision with a Multi-part Mirror. Lecture Notes in Artificial Intelligence, Vol. 2019. Springer-Verlag, Berlin Heidelberg New York (2000) 179–188

[8] Nakamura, T., Ohhara, M., Ogasawara, T., Ishiguro, H.: Fast Self-Localization Method for Mobile Robots Using Multiple Omnidirectional Vision Sensors. IPSJ SIG Notes on CVIM. Vol. 2001, No.125-18. Information Processing Society of Japan (2001) 133–138 (In Japanese)

[9] Geist, A., et al.: PVM : Parallel Virtual Machine. A Users'Guide and Tutorial for Networked Parallel Computing, The MIT Press (1994) ftp://ftp.netlib.org/pvm3/book/pvm-book.ps

[10] Egami, K., Yagi, Y., Yachida, M.: Generation of Environmental Map by Multiple Image Sensing System. IPSJ SIG Notes on CV. Vol. 1995, No.92-8. Information Processing Society of Japan (1995) 57–64 (In Japanese)

[11] Jamzad, M., Foroughnassiraei, A., Chiniforooshan, E., Ghorbani, R., Kazemi, M., Chitsaz, H., Mobasser, F., Sadjad, S.B.: Middle Sized Soccer Robots: ARVAND. Lecture Notes in Artificial Intelligence, Vol. 1856. Springer-Verlag, Berlin Heidelberg New York (1999) 61–73

[12] Bandlow, T., Klupsch, M., Hanek, R., Schmitt, T.: Fast Image Segmentation, Object Recognition and Localization in a RoboCup Scenario. Lecture Notes in Artificial Intelligence, Vol. 1856. Springer-Verlag, Berlin Heidelberg New York (1999) 174–185

〒162 東京都新宿区津久戸町 4-7 OS ビル 402 号室 03-5261-3401 Fax: 03-5261-3402

(本研究会についてのお問い合わせは下記にお願いします.)

ＡＩチャレンジ研究会
主 査
奥乃 博
京都大学大学院 情報学研究科 知能情報学専攻 /
科学技術振興事業団 ERATO
北野共生システムプロジェクト
〒150-0001 東京都渋谷区神宮前 6-31-15
マンション 31, 6A 室
03-5468-1661 Fax: 03-5468-1664
okuno@nue.org

幹 事
浅田 稔
大阪大学大学院 工学研究科
知能・機能創成工学専攻
武田 英明
国立情報学研究所 知能システム研究系
樋口 哲也
独立行政法人 産業技術総合研究所
田所 諭
神戸大学 工学部 情報知能工学科

Executive Committee
Chair
Hiroshi G. Okuno
Dept. of Intelligence Science and Technology,
Gradulate School of Informatics
Kyoto University/
Kitano Symbiotic Systems Project,
ERATO, JST
Manshon 31, Room 6A
6-31-15 Jingumae, Shibuya, Tokyo
150-0001 JAPAN

Secretary
Minoru Asada
Dept. of Information and Intelligent Engineering
Graduate School of Engineering
Osaka University
Hideaki Takeda
National Institute of Informatics
Tetsuya Higuchi
National Institute of Advanced Industrial Science and Technology
Satoshi Tadokoro
Dept. of Information and Intelligent Engineering
Kobe University

SIG-AI-Challenges home page (WWW): `http://www.symbio.jst.go.jp/SIG-Challenge/`