Applied Mathematical Sciences, Vol. 6, 2012, no. 116, 5775 - 5780

Computational Experiments for the Problem of Selection of a Minimal Set of Visual Landmarks

Anna Gorbenko

Department of Intelligent Systems and Robotics
Ural Federal University
620083 Ekaterinburg, Russia
gorbenko.ann@gmail.com

Vladimir Popov

Department of Intelligent Systems and Robotics
Ural Federal University
620083 Ekaterinburg, Russia
Vladimir.Popov@usu.ru

Abstract

In this paper we consider an approach to solve the problem of selection of a minimal set of landmarks. Our approach is based on constructing intelligent algorithms to solve logical models for the problem.

Mathematics Subject Classification: 68T40

Keywords: visual landmarks, landmarks selection, **NP**-complete

Intelligent algorithms for visual navigation is extensively used in contemporary robotics and other vision systems (see e.g. [1, 2, 3, 4, 5, 6, 7]). Using systems of visual landmarks has been widely applied for mobile robot navigation (see e.g. [8, 9, 10]). In this paper we consider the problem of selection of a minimal set of visual landmarks SMSL [10].

In papers [11, 12, 13, 14, 15] the authors considered some algorithms to solve different logical models (see also [16, 17, 18, 19, 20, 21]). In this paper we consider reductions from SMSL-D (see [10]) to SAT and 3SAT.

Let

$$\varphi = \wedge_{1 \leq t \leq d} \vee_{1 \leq i \leq k} x[t, i],$$

$$\psi = \wedge_{1 \le t \le d} \wedge_{1 \le i[1] < i[2] \le k} (\neg x[t,i] \vee \neg x[t,i[2]]),$$

$$\tau[j,l] = \vee_{1 \le t \le d, 1 \le i \le k, F_j[l] \in \mathcal{L}[i,j]} x[t,i],$$

$$\tau = \wedge_{1 \le j \le n} \wedge_{1 \le l \le m_j, F_j[l] \in \cup_{L_i \in \mathcal{L}} L_i} \tau[j,l],$$

$$\xi = \varphi \wedge \psi \wedge \tau.$$

Theorem. *Given a set of landmarks*

$$\mathcal{L} = \{L_1, L_2, \ldots, L_k\}$$

and a positive integer d. *There is a set* $\mathcal{S} \subseteq \mathcal{L}$ *such that* $\cup_{L_i \in \mathcal{S}} L_i = \cup_{L_i \in \mathcal{L}} L_i$ *and* $|\mathcal{S}| \le d$ *if and only if* ξ *is satisfiable.*

Proof. Suppose that there is a set $\mathcal{S} \subseteq \mathcal{L}$ such that $\cup_{L_i \in \mathcal{S}} L_i = \cup_{L_i \in \mathcal{L}} L_i$ and $|\mathcal{S}| \le d$. Without loss of generality we can assume that $|\mathcal{S}| = d$. Let

$$\mathcal{S} = \{S_1, S_2, \ldots, S_d\}.$$

Since $|\mathcal{S}| = d$, we can suppose that $x[t,i] = 1$ if and only if $S_t = L_i$. It is easy to see that $\varphi = 1$ if and only if, for any t, there is at least one value of i such that $x[t,i] = 1$. Respectively, $\psi = 1$ if and only if, for any t, there is no more than one value of i such that $x[t,i] = 1$. Therefore, by definition of $x[t,i]$, $\varphi \wedge \psi = 1$. In view of $\cup_{L_i \in \mathcal{S}} L_i = \cup_{L_i \in \mathcal{L}} L_i$, it is easy to check that $\tau = 1$. So, $\xi = 1$.

Suppose now that $\xi = 1$. Since $\varphi \wedge \psi = 1$ if and only if, for any t, there is only one value of i such that $x[t,i] = 1$, we can consider values of $x[t,i]$ as a definition of $\mathcal{S}$. In particular, $L_i \in \mathcal{S}$ if and only if there exist t such that $x[t,i] = 1$. Note that $\tau = 1$ if and only if, for any $j \in \{1, 2, \ldots, n\}$ and $l \in \{1, 2, \ldots, m_j\}$ such that $F_j[l] \in \cup_{L_i \in \mathcal{L}} L_i$, there exist t and i such that $x[t,i] = 1$ where $F_j[l] \in \mathcal{L}[i,j]$. Thus, by definition of $\mathcal{S}$, for any $j \in \{1, 2, \ldots, n\}$ and $l \in \{1, 2, \ldots, m_j\}$ such that $F_j[l] \in \cup_{L_i \in \mathcal{L}} L_i$, there exist i such that $L_i \in \mathcal{S}$ and $F_j[l] \in \mathcal{L}[i,j]$. Therefore, $\cup_{L_i \in \mathcal{S}} L_i = \mathcal{F}$. □

It is easy to check that ξ is a CNF. So, ξ gives us an explicit reduction from SMSL-D to SAT. By direct verification we can check that

$$\begin{aligned}
\alpha \quad &\Leftrightarrow \quad (\alpha \vee \beta_1 \vee \beta_2) \wedge \\
&\qquad (\alpha \vee \neg\beta_1 \vee \beta_2) \wedge \\
&\qquad (\alpha \vee \beta_1 \vee \neg\beta_2) \wedge \\
&\qquad (\alpha \vee \neg\beta_1 \vee \neg\beta_2), && (1) \\
\vee_{j=1}^{l} \alpha_j \quad &\Leftrightarrow \quad (\alpha_1 \vee \alpha_2 \vee \beta_1) \wedge \\
&\qquad (\wedge_{i=1}^{l-4} (\neg\beta_i \vee \alpha_{i+2} \vee \beta_{i+1})) \wedge \\
&\qquad (\neg\beta_{l-3} \vee \alpha_{l-1} \vee \alpha_l), && (2) \\
\alpha_1 \vee \alpha_2 \quad &\Leftrightarrow \quad (\alpha_1 \vee \alpha_2 \vee \beta) \wedge
\end{aligned}$$

$$\begin{aligned} & (\alpha_1 \vee \alpha_2 \vee \neg\beta), & (3) \\ \vee_{j=1}^{4}\alpha_j \quad \Leftrightarrow \quad & (\alpha_1 \vee \alpha_2 \vee \beta_1) \wedge \\ & (\neg\beta_1 \vee \alpha_3 \vee \alpha_4) & (4) \end{aligned}$$

where $l > 4$. Using relations (1) – (4) we can easily obtain an explicit transformation ξ into ζ such that $\xi \Leftrightarrow \zeta$ and ζ is a 3-CNF. It is clear that ζ gives us an explicit reduction from SMSL-D to 3SAT.

We obtain explicit reductions from SMSL-D to SAT and 3SAT. We use algorithms fgrasp and posit from [22]. Also, we design our own genetic algorithm for SAT which based on algorithms from [22].

Consider a boolean function

$$g(x_1, x_2, \ldots, x_n) = \wedge_{i=1}^{m}\mathcal{C}_i,$$

where $m \geq 1$, and each of the $\mathcal{C}_i$ is the disjunction of one or more literals. Let $|\mathcal{C}_i|$ be a number of literals in $\mathcal{C}_i$. Let $occ(x_i, g)$ be a number of occurrences of x_i in g. Respectively, let $occ(\neg x_i, g)$ be a number of occurrences of x_i in g. For example, if

$$g = (x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_4) \wedge (\neg x_1 \vee x_5),$$

then $occ(x_1, g) = 2$, $occ(\neg x_1, g) = 1$.

We consider a number of natural principles that define importance of a variable x_i for satisfiability of boolean function g. These principles suggest us correct values of variables.

1. If $occ(x_i, g) \geq 0$ and $occ(\neg x_i, g) = 0$, then $x_i = 1$.

2. If $occ(x_i, g) = 0$ and $occ(\neg x_i, g) \geq 0$, then $x_i = 0$.

3. If $x_i = \mathcal{C}_j$ for some j, then $x_i = 1$.

4. If
$$\min_{occ(x_i, \mathcal{C}_j) > 0} |\mathcal{C}_j| \leq \min_{occ(\neg x_i, \mathcal{C}_j) > 0} |\mathcal{C}_j|,$$
then $x_i = 1$.

5. Given positive integers
$$p_1, p_2, \ldots, p_m, q_1, q_2, \ldots, q_m$$
and a set of rational numbers
$$\{\alpha_{i,u}, \beta_{i,v} \mid 1 \leq i \leq m, 1 \leq u \leq p_i, 1 \leq v \leq q_i\}.$$

If

$$\sum_{1 \le j \le m, 1 \le u \le p_j, |\mathcal{C}_j| = u} \alpha_{j,u} occ(x_i, \mathcal{C}_j) \ge \sum_{1 \le j \le m, 1 \le v \le q_j, |\mathcal{C}_j| = v} \beta_{j,v} occ(\neg x_i, \mathcal{C}_j),$$

then $x_i = 1$.

Based on these principles, we can consider the following five types of commands: $P_1, P_2, \ldots, P_5$. Also we consider the following three commands for run algorithms: Try_fgrasp, Try_posit, and Try_ga, where Try_ga runs a simple genetic algorithm.

Denote by $\mathcal{R}$ the set of commands of these eight types. Arbitrary element of $\mathcal{R}^*$ it is possible to consider as a program for finding values of variables of a boolean function. We assume that such programs are executed on a cluster.

Execution of each of commands of type P_i reduces the number of variables of a boolean function by one. Execution of each of commands Try_fgrasp, Try_posit, and Try_ga consists in the run of corresponding algorithm for current boolean function on a separate set of calculation nodes and the transition to the next command.

Algorithms fgrasp and posit we run only on one calculation node. Genetic algorithms can be used in parallel execution. We use auxiliary genetic algorithm which determine the number of calculation nodes.

Initially, we selected a random subset of $\mathcal{R}^*$. We use a genetic algorithm to select a program from the current subset of $\mathcal{R}^*$ and a genetic algorithm for evolving the current subset of $\mathcal{R}^*$. The evolution of the current subset of $\mathcal{R}^*$ implemented on a separate set of calculation nodes. For every subsequent boolean functions it is used the current subset of $\mathcal{R}^*$ which is obtained by taking into account the results of previous runs.

We use heterogeneous cluster based on three clusters (Cluster USU, Linux, 8 calculation nodes; umt, Linux, 256 calculation nodes; um64, Linux, 124 calculation nodes) [23].

Algorithms fgrasp and posit used only for 3SAT. For SAT used simple genetic algorithm (SGA), and our algorithm (OA). Selected experimental results are given in Tables 1, 2.

References

[1] A. Gorbenko, A. Lutov, M. Mornev, and V. Popov, Algebras of Stepping Motor Programs, *Applied Mathematical Sciences*, 5 (2011), 1679-1692.

[2] A. Gorbenko, V. Popov, and A. Sheka, Robot Self-Awareness: Temporal Relation Based Data Mining, *Engineering Letters*, 19 (2011), 169-178.

time	fgrasp	posit	SGA	OA
average	24.1 min	24.7 min	37.8 min	19.7 min
maximum	14.7 h	15.6 h	26.4 h	7.2 h
best	4.6 min	4.1 min	43 sec	52 sec

Table 1: Experimental results for reduction to 3SAT.

time	SGA	OA
average	28.8 min	23.7 min
maximum	22.4 h	20.6 h
best	37 sec	43 sec

Table 2: Experimental results for reduction to SAT.

[3] A. Gorbenko and V. Popov, On Face Detection from Compressed Video Streams, *Applied Mathematical Sciences*, 6 (2012), 4763-4766.

[4] A. Gorbenko and V. Popov, Usage of the Laplace Transform as a Basic Algorithm of Railroad Tracks Recognition, *International Journal of Mathematical Analysis*, 6 (2012), 2413-2417.

[5] A. Gorbenko and V. Popov, Robot Self-Awareness: Occam's Razor for Fluents, *International Journal of Mathematical Analysis*, 6 (2012), 1453-1455.

[6] A. Gorbenko and V. Popov, The Force Law Design of Artificial Physics Optimization for Robot Anticipation of Motion, *Advanced Studies in Theoretical Physics*, 6 (2012), 625-628.

[7] A. Gorbenko and V. Popov, Self-Learning Algorithm for Visual Recognition and Object Categorization for Autonomous Mobile Robots, *Proceedings of the CICA 2011*, (2012), 1289-1295.

[8] A. Gorbenko and V. Popov, On the Problem of Placement of Visual Landmarks, *Applied Mathematical Sciences*, 6 (2012), 689-696.

[9] A. Gorbenko and V. Popov, A Real-World Experiments Setup for Investigations of the Problem of Visual Landmarks Selection for Mobile Robots, *Applied Mathematical Sciences*, 6 (2012), 4767-4771.

[10] A. Gorbenko and V. Popov, The Problem of Selection of a Minimal Set of Visual Landmarks, *Applied Mathematical Sciences*, 6 (2012), 4729-4732.

[11] A. Gorbenko, M. Mornev, and V. Popov, Planning a Typical Working Day for Indoor Service Robots, *IAENG International Journal of Computer Science*, 38 (2011), 176-182.

[12] A. Gorbenko and V. Popov, Programming for Modular Reconfigurable Robots, *Programming and Computer Software*, 38 (2012), 13-23.

[13] A. Gorbenko and V. Popov, On the Optimal Reconfiguration Planning for Modular Self-Reconfigurable DNA Nanomechanical Robots, *Advanced Studies in Biology*, 4 (2012), 95-101.

[14] A. Gorbenko and V. Popov, The set of parameterized k-covers problem, *Theoretical Computer Science*, 423 (2012), 19-24.

[15] A. Gorbenko, V. Popov, and A. Sheka, Localization on Discrete Grid Graphs, *Proceedings of the CICA 2011*, (2012), 971-978.

[16] A. Gorbenko and V. Popov, The Longest Common Parameterized Subsequence Problem, *Applied Mathematical Sciences*, 6 (2012), 2851-2855.

[17] A. Gorbenko and V. Popov, The Binary Paint Shop Problem, *Applied Mathematical Sciences*, 6 (2012), 4733-4735.

[18] A. Gorbenko and V. Popov, The Longest Common Subsequence Problem, *Advanced Studies in Biology*, 4 (2012), 373-380.

[19] A. Gorbenko and V. Popov, Element Duplication Centre Problem and Railroad Tracks Recognition, *Advanced Studies in Biology*, 4 (2012), 381-384.

[20] A. Gorbenko, M. Mornev, V. Popov, and A. Sheka, The problem of sensor placement for triangulation-based localisation, *International Journal of Automation and Control*, 5 (2011), 245-253.

[21] A. Gorbenko, M. Mornev, V. Popov, and A. Sheka, The Problem of Sensor Placement, *Advanced Studies in Theoretical Physics*, 6 (2012), 965-967.

[22] SATLIB — The Satisfiability Library. [Online]. Available: http://people.cs.ubc.ca/~hoos/SATLIB/index-ubc.html

[23] Web page "Computational resources of IMM UB RAS". (In Russian.) [Online]. Available: http://parallel.imm.uran.ru/mvc_now/hardware/supercomp.htm

Received: July, 2012