GERAÇÃO DE MAPAS TOPOLÓGICOS NÃO-DETERMINÍSTICOS EM ROBÓTICA MÓVEL VIA MODELOS ESCONDIDOS DE MARKOV EM CASCATA

Ederson R. Wagner*, Carlos H. C. Ribeiro*

*Laboratório de Navegação e Controle de Robôs Móveis Autônomos
Instituto Tecnológico de Aeronáutica
Pça. Mal. Eduardo Gomes, 50 - Vila das Acácias 12228-900 - São José dos Campos - SP

Emails: wagner@ita.br, carlos@ita.br

Abstract— The task of building high quality topological maps to mobile robots through the use Hidden Markov Models can be difficult to achieve in situations where there is much uncertainty in the robot's sensors or there are navigation constraints that prevent the acquisition of an appropriate set of sensor readings. Low quality sensorial systems can generate a high amount of possible observation symbols that are used as input data in the training process of Hidden Markov Models. The amount of information available about the environment directly affects the system, that becomes sensitive to observation ambiguities for each state. Hidden Markov Models cascade, that uses segmented sensorial data, reduces the number of possible observations per corresponding state, what represents a better solution regarding information ambiguities, hence generating a higher quality map, closer to the real one. In this paper, a procedure based on Hidden Markov Models cascade to generate high quality topological maps, compared to those created by simple monolithic Hidden Markov Models, is discussed.

Keywords— Mobile Robotics, Topological Maps, Hidden Markov Models.

Resumo— A geração de mapas topológicos de qualidade, para robôs móveis, utilizando Modelos Escondidos de Markov, é muito difícil de ser conseguida caso o sistema de sensoriamento do robô possua muitas incertezas ou existem limitações de navegação no sentido de não conseguir obter um conjunto de leituras sensoriais de tamanho apropriado. Sistemas de sensoriamentos de baixa qualidade causam um enorme conjunto de possíveis símbolos de observação que são usados como entrada no treinamento dos Modelos Escondidos de Markov. O fato de não existir um conjunto suficientemente grande de informações relativas ao ambiente torna o sistema muito frágil no sentido de ambigüidades nas observações para cada estado. O cascateamento dos Modelos Escondidos de Markov, através da segmentação das informações sensoriais, reduz o número de possíveis observações para cada estado correspondente e consegue resolver de melhor forma o problema de ambigüidades gerando um mapa de qualidade superior, ou seja, mais próximo do ambiente real. Nesse trabalho é investigado um procedimento baseado em cascateamento de Modelos Escondidos de Markov com a finalidade de encontrar um mapa topológico de qualidade superior ao mapa gerado pelo procedimento básico baseado no treinamento de Modelos Escondidos de Markov monolíticos.

Keywords— Robótica Móvel, Mapas Topológicos, Modelos Escondidos de Markov.

1 Introdução

Modelos Escondidos de Markov (MEMs), também chamados de Modelos Ocultos de Markov, representam uma variedade de sistemas dinâmicos não determinísticos, tais como sistemas probabilísticos de transições entre estados com observações e estados discretos. Cada estado no modelo está associado a uma distribuição, ou a uma função densidade de probabilidade em relação às possíveis observações. As incertezas dinâmicas do sistema modelado são representadas pelas probabilidades de transição entre os estados do modelo. A cada estado é atribuída uma distribuição de probabilidade sobre os possíveis estados posteriores (Shatkay, 1998).

Na geração de mapas para robôs móveis, os MEMs possibilitam a geração de mapas topológicos das possíveis transições entre os estados do robô baseando-se somente nas leituras sensoriais adquiridas em cada estado (Shatkay and Kaelbling, 1997) e (Marceau et al., 2002). Um estado não necessariamente corresponde à postura do robô em um mapa, e vários outros atributos (variáveis de estado) podem estar envolvidos na representação do estado, como por exemplo, a carga da bateria do robô, se ele está ou não portando um objeto (caso o robô possua um sistema de garra), etc.

Este artigo analisa uma arquitetura de MEMs em cascata (Wagner and Ribeiro, 2006) para geração de mapas topológicos de ambientes não-determinísticos. Essa arquitetura permite encontrar um mapa de maior qualidade no sentido de melhor representar as possíveis transições do robô. A arquitetura de MEMs em cascata permite ainda utilizar um número maior de informações sensoriais (resolvendo o problema de ambigüidade entre estados) bem como trabalhar com um conjunto menor de observações.

Na Seção 2 é apresentada a arquitetura básica de MEMs, bem como os algoritmos de treinamento do modelo. A Seção 3 aborda a arquitetura de MEMs em cascata utilizada. Na Seção 4 são apresentados os experimentos realizados, e os resultados obtidos são mostrados e comentados na subseção 4.1. Finalmente, a Seção 5 apresenta alguns comentários a respeito do uso da arquitetura de MEMs em cascata, bem como linhas de pesquisa futuras.

2 Os Modelos Escondidos de Markov e seus Algoritmos

Conforme (Rabiner, 1989), (Britto, 2001) e (Shatkay and Kaelbling, 1997), os MEM são representações utilizadas para se modelar um sinal através de uma seqüência de observações. Em uma Cadeia de Markov supõe-se uma fonte gerando tais saídas observáveis, denominada de Fonte de Markov. Os símbolos gerados a partir dessa fonte são dependentes apenas de observações anteriores, as quais foram geradas da mesma forma e assim sucessivamente.

Cada estado de uma Cadeia de Markov representa portanto uma observação/símbolo de um evento físico correspondente, e o objetivo é computar a partir de uma dada seqüência de símbolos quais foram os estados que geraram tal seqüência. Contudo, em aplicações reais mais de um símbolo pode ser observado por estado, e a origem dessas observações torna-se imprecisa.

2.1 Definições

Um MEM pode ser descrito por $\lambda = (Q, V, A, B, \Pi, T)$, onde:

- T é comprimento da seqüência de observações,
- $Q = q_1, q_2, ..., q_N$ é o conjunto de estados do modelo, Q_t designa o estado da cadeia no instante t, e N é o número de estados do modelo,
- $V = v_1, v_2, ..., v_M$ é o conjunto de símbolos de observações possíveis. O_t designa o símbolo da observação no instante t e M é o número de símbolos da observação.
- $A = \{a_{ij}\} 1 \leq i, j \leq N$ é a matriz de transição de estados. O elemento a_{ij} corresponde à probabilidade de transição do estado q_i para o estado q_j:

$$a_{ij} = P(Q_{t+1} = q_j \mid Q_t = q_i) \qquad (1)$$

com as seguintes restrições:

$$a_{ij} \geq 0 \quad \forall\, i, j \quad e \quad \sum_{j=1}^{N} a_{ij} = 1 \quad \forall\, i \quad (2)$$

- $B = \{b_{jk}\} 1 \leq j \leq N;\ 1 \leq k \leq M$ é a matriz de probabilidades de observação de símbolos condicionadas pelos estados da cadeia. O elemento b_{jk} representa a probabilidade de observar o símbolo V_k quando o modelo se encontra no estado q_j:

$$b_{jk} = P(O_t = v_k | Q_t = q_j) 1 \leq j \leq N 1 \leq k \leq M \qquad (3)$$

com as seguintes restrições:

$$b_{jk} \geq 0 \quad \forall\, j, k \quad e \quad \sum_{k=1}^{M} b_{jk} = 1 \quad \forall\, j \quad (4)$$

- $\Pi = \{\pi_i\}\ 1 \leq i \leq N$ é o conjunto de densidades de probabilidade inicial, π_i representa a probabilidade de que o processo de Markov inicie no estado q_i:

$$\pi_i = P(Q_1 = q_i) \quad 1 \leq i \leq N \qquad (5)$$

$$\pi_i \geq 0 \quad \forall i \quad e \quad \sum_{j=1}^{N} \pi_i = 1 \qquad (6)$$

2.2 Treinamento de um HMM

O treinamento de um modelo estatístico não é algo simples de se fazer. Por ser um problema de otimização contínua de várias variáveis (parâmetros A, B e Π do modelo λ), pode ser necessário aplicar-se um método de otimização baseado no gradiente. Uma das melhores opções neste caso é um tipo específico de algoritmo EM (*Expectation-Maximization*) denominado Algoritmo de Baum-Welch (também conhecido como Algoritmo *Forward-Backward*), que busca treinar os modelos por máxima verossimilhança.

Devido ao grande número de fatores de variabilidade presentes em mapas topológicos, a vantagem deste algoritmo é que ele considera todas as possíveis combinações de estados prováveis e maximiza a verdadeira probabilidade de emissão das observações (Baum et al., 1970).

Apesar de não haver garantias de que o Algoritmo de Baum-Welch irá convergir para um máximo global, sua convergência a um bom máximo local ocorre, em geral, muito rapidamente.

O Algoritmo de Baum-Welch consiste dos seguintes passos:

1. Definir um modelo inicial λ_0
2. Computar o novo λ baseando-se em λ_0 e as observações O (Passos E e M do Algoritmo EM)
3. Se $\log P(O|\lambda) - \log P(O|\lambda_0) < \Delta$ pare
4. Senão $\lambda_0 := \lambda$ e retorna ao passo 2.

Definindo-se

- $\xi(i, j)$ a probabilidade de estar no estado i no tempo t e no estado j no tempo $t + 1$:

$$\xi(i,j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1} \beta_{t+1}(j)}{P(O|\lambda)} = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1} \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)} \qquad (7)$$

- $\gamma_t(i)$ a probabilidade de estar no estado i no tempo t, dada a seqüência de observações:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \tag{8}$$

a atualização do modelo λ acontece da seguinte forma:

- $\hat{\pi}_i$ = número estimado de vezes em que foi encontrado o estado i no tempo $(t = 1)$
- $\hat{a}_{ij}$ = (número estimado de transições do estado i ao estado j) / (número estimado de transições a partir do estado i):

$$\hat{a}_{ij} = \frac{\sum \xi_t(i,j)}{\sum \gamma_t(i)} \tag{9}$$

- $\hat{b}_j(k)$ = (número estimado de vezes que esteve-se no estado j e observou-se o símbolo k) / (número de vezes que esteve-se no estado j):

$$\hat{b}_j(k) = \frac{\sum_{t,o_t=k} \gamma_t(j)}{\sum_t \gamma_j} \tag{10}$$

onde, $\sum_{t=1}^{T} \gamma_t(i)$ é o número estimado de vezes que o estado i é visitado e $\sum_{t=1}^{T-1} \xi_t(i,j)$ é o número estimado de transições do estado i ao estado j

O interesse do algoritmo de Baum-Welch reside no fato de que as fórmulas de reestimação de A, B e Π acima garantem o incremento de $P(O|\lambda)$ até que um ponto de convergência seja atingido. Para maiores detalhes, vide (Rabiner, 1989).

3 Cascateamento de MEMs

Como em qualquer algoritmo de reconhecimento, no uso de MEMs a qualidade do sistema de sensoriamento influencia diretamente na qualidade do modelo final obtido. Esta qualidade pode ser verificada pela quantidade de diferentes símbolos observação (M) em relação ao número de estados do modelo (N).

O tamanho de M também é afetado pelo fato de existirem muitas leituras sensoriais para cada estado. Por exemplo, considere 32 sonares e um *scanner* a laser que varre 180°para um modelo de 4 estados, nesse caso, existirão 212 diferentes informações sensoriais para cada estado (32 dos sonares mais 180 do laser). Sendo que cada símbolo de observação em um MEM representa uma configuração única de leituras nos sensores, o número de possíveis símbolos de observação será enorme.

A situação ainda piora quando não se tem um vetor de observações (T) informativo o suficiente para resolver problemas de ambigüidade de estados decorrentes das leituras sensoriais o que, conseqüentemente, pode gerar um mesmo símbolo de observação para diferentes estados do modelo.

A Figura 1 mostra o modelo básico de treinamento/aprendizado de MEMs onde o conjunto de leituras sensoriais pode corresponder, por exemplo, à leitura de 10 sonares distribuídos em um robô móvel, os ângulos e distâncias lidos por um *scanner* a *laser*, etc.

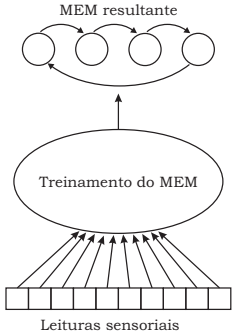

Figura 1: Arquitetura básica de geração de MEMs

O treinamento do MEM é feito então pelo algoritmo de Baum-Welch apresentado na Seção 2, e consiste em encontrar o modelo que melhor representa a seqüência de estados "escondidos"no conjunto de leituras sensoriais.

3.1 MEMs em Cascata

Neste trabalho será considerada uma arquitetura de MEMs em cascata para gerar um mapa topológico de um ambiente com transições não-determinísticas entre os estados. A principal vantagem do uso do cascateamento, nessa aplicação, é a possibilidade de trabalhar com um menor número de símbolos de observação em cada sub-camada da arquitetura, conseguindo assim, gerar MEMs mais completos e confiáveis e que representam melhor as transições entre os estados do robô.

A redução de M é obtida pela divisão do conjunto de leituras sensoriais e geração de MEMs independentes e arranjados segundo níveis hierárquicos para cada conjunto de leituras parciais. Ao invés de tentar gerar o modelo de transição de estados baseando-se em um conjunto de observação composto por K elementos (sonares, leituras de laser, etc.) que geram um M de maior amplitude, definem-se dois (ou mais) sub-modelos através de conjuntos de observações sensoriais com um tamanho reduzido, por exemplo $K/2$ elementos, e assim por diante. Posteriormente encontra-se o MEM que corresponde à distribuição real (de nível mais alto) através de leituras geradas pelos modelos de nível mais baixo.

A saída dos modelos do nível inferior serão usadas como entrada no modelo de nível mais alto, reduzindo assim o número de leituras sensoriais de

entrada deste modelo.

Formalmente, a topologia de um MEM em cascada é mostrada na Figura 2, isto é, seja o modelo em cascata $\Lambda^u = \{\Lambda^{l_1}, \Lambda^{l_2}, \Lambda^u\}$ onde cada componente dos modelos Λ^{l_j} e Λ^u também são MEMs definidos de acordo coma a descrição na Seção 2. Seja $\{Q_n^{l_1}, V_n^{l_1}\}$, $\{Q_n^{l_2}, V_n^{l_2}\}$ e $\{Q_n^u, V_n^u\}$ os estados e seqüências de observações que correspondem, respectivamente, a Λ^{l_1}, Λ^{l_2} e Λ^u. Neste modelo, as observações $V_k^{l_j}$ dos modelos de nível inferior Λ^{l_j} são observações geradas pelas leituras sensoriais. As observações V_k^u do modelo de nível superior Λ^u são geradas pela seqüência de estados dos modelos de nível inferior, ou seja, $V_k^u = (Q_k^{l_1}, Q_k^{l_2})$, e Λ^u modela a distribuição conjunta de $Q_k^{l_1}$ e $Q_k^{l_2}$ para cada estado $j = 1, ..., N$.

A Figura 2 apresenta a estrutura da arquitetura de MEMs em cascata.

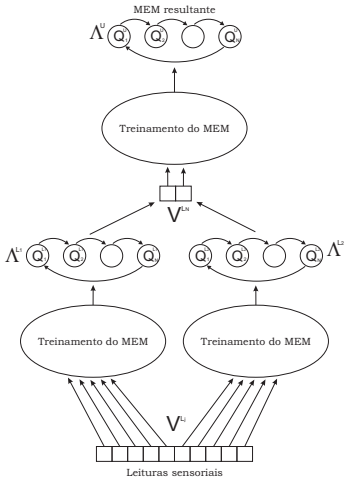

Figura 2: Arquitetura em cascata para geração de MEMs

No modelo em cascata, os MEMs do nível inferior são, portanto, treinados a partir de leituras sensoriais com menor número de elementos, diminuindo assim o número de símbolos de entrada (M). Então, através das probabilidades de transição entre os estados encontrados nos modelos de nível inferior é possível replicar a seqüência de estados observada no vetor de leituras sensoriais e usar essa nova seqüência gerada como entrada do nível superior. O número de estados (N) dos MEMs de nível inferior deve ser obrigatoriamente o mesmo do número de estados do nível superior.

Conforme observado na Figura 2, o tamanho do vetor de leituras sensoriais passa de 10 no MEM básico para 5 em cada um dos MEMs de nível inferior, e para 2 para o MEM de nível superior.

É possível que esta arquitetura não seja interessante quando se possui um vetor de leituras sensoriais muito pequeno, pois o treinamento e geração de vários MEMs consome um tempo muito grande e provavelmente irá gerar um modelo tão bom quanto ao gerado por um único MEM em sua forma mais simples.

4 Experimentos Realizados

Com a finalidade de avaliar o funcionamento da arquitetura de MEMs em cascata, foi realizada uma bateria de testes simulados em ambiente *MatLab*© treinando vários MEMs em sua arquitetura básica e em cascata para diferentes tamanhos de vetores de leitura sensorial (V) e de seqüência de observações (T).

As leituras sensoriais foram geradas a partir de um conjunto de tamanho V de sensores simulando um robô navegando em um ambiente de N estados com probabilidades de transição conhecidas. Posteriormente, foi inserido às leituras "limpas"geradas, um ruído com distribuição normal ($N(0, 1)$). Estas informações simulam um vetor de leitura de sonares, *scanner* a laser ou outros sensores como, por exemplo, a carga da bateria do robô.

Para todos os casos foi utilizado um modelo de quatro estados ($N = 4$) com probabilidades de transição entre os estados conforme mostra a Figura 3. As matrizes A, B e PI dos MEMs envolvidos foram inicializadas de acordo com uma distribuição uniforme para cada um dos treinamentos.

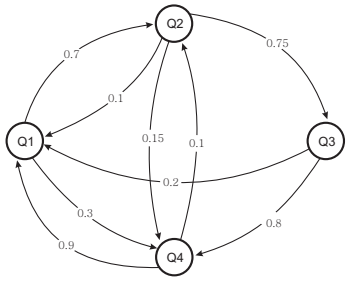

Figura 3: Modelo utilizado para os experimentos onde os nós representam os Estados do modelo e as arestas as probabilidades de Transição entre eles.

Cada uma das configurações foi treinada por 100 vezes com um conjunto de leituras sensoriais idênticas para ambas arquiteturas (básica e em cascata). O vetor de leituras sensoriais (V) teve sua variação entre 2 e 32 diferentes leituras (simulando o uso de 2 a 32 sonares, leituras de um *scanner* a laser ou até mesmo diferentes sensores simultaneamente) e o tamanho da seqüência de observações(T) variou entre 100 e 800 passos, simulando um robô movendo-se no ambiente e a cada estado encontrado coletando um conjunto de leituras sensoriais (V) até atingir o valor de T passos.

A coluna à esquerda da Tabela 1 apresenta todas as configurações de MEMs em cascata

testados e os dos resultados apresentados nas outras colunas são melhor discutidos na Seção 4.2).

A Figura 4 ilustra o padrão de configuração para MEMs em cascata.

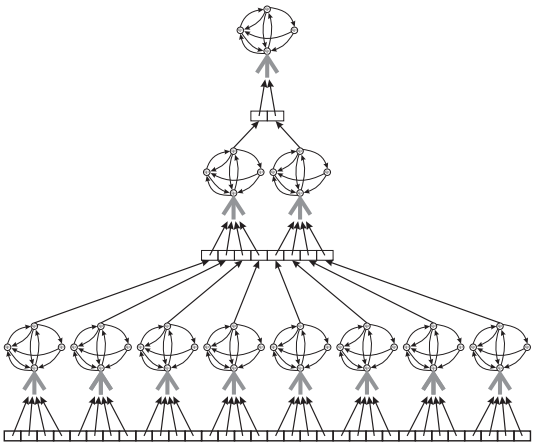

Figura 4: Exemplo da configuração de MEMs em cascata $MC = (8, 2, 1 - 32, 8, 2)$ onde os primeiros números $(8, 2, 1)$ representam o número de MEMs (W) em cada nível e o outro conjunto de números $(32, 8, 2)$ representam o número de informações sensoriais (V) para cada nível

Para encontrar esses valores médios de V e T foram realizados testes preliminares com uma grande variedade de outras configurações, porém foi nessa faixa de valores em que começam a ser evidenciadas as diferenças entre o MEM em sua configuração básica e o MEM em cascata.

4.1 Avaliação dos Resultados

Os resultados foram avaliados de duas formas distintas: a) pelo tempo gasto para treinar os MEMs e b) pela qualidade do mapa gerado, obtida através da divergência de Kulback-Leibler(KL). A divergência de KL é uma medida que quantifica a variação média da informação associada entre duas distribuições de probabilidades e quanto maior o valor de KL maior a diferença entre as distribuições. Para maiores detalhes, vide (Kullback and Leibler, 1951).

4.2 Resultados Obtidos

Dentre todos os experimentos realizados, em média, o modelo em cascata, independente de sua configuração, mostrou-se melhor em relação ao modelo básico em relação à divergência de KL. Porém, o tempo de treinamento está ligado ao número de MEMs envolvidos no processo. Um bom exemplo do tempo envolvido é o modelo apresentado na Figura 4. Nesse caso, são treinados 11 MEMs, portanto, o tempo será, em média, 11 vezes maior para a arquitetura em cascata. Além disso, os modelos dos níveis inferiores devem servir como geradores de informações para os MEMs de seu nível superior, isso implica em um acréscimo ainda maior na variável tempo.

A Figura 5 apresenta essa relação, onde é claramente percebida a enorme diferença no tempo de treinamento para o modelo em cascata em relação à arquitetura básica de MEMs. Percebe-se também a enorme diferença na divergência de KL, onde a arquitetura em cascata gerou um mapa de melhor qualidade em relação à arquitetura básica.

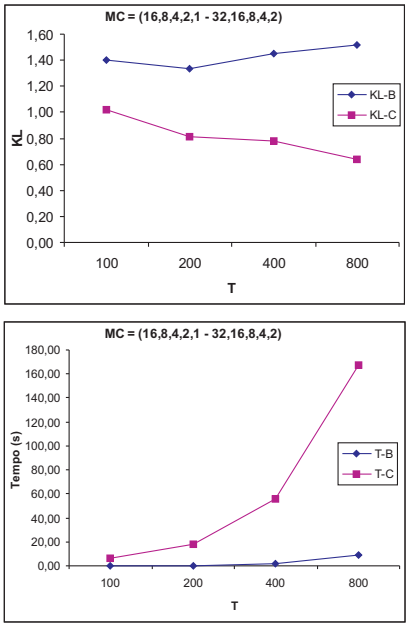

Figura 5: Divergência de KL e Tempo para o Modelo $MC = (16, 8, 4, 2, 1 - 32, 16, 8, 4, 2)$ e T variando entre 100 e 800

A Tabela 1 apresenta a média para 100 realizações de diferentes arquiteturas de MEMs, conforme mostra a primeira coluna. As colunas restantes referem-se a KL para T variando entre 100 e 800 passos.

A Figura 6 mostra a média da divergência de KL para todos os experimentos realizados onde $T = 800$. Mais uma vez fica clara a superioridade da arquitetura de MEMs em cascata no sentido de encontrar o melhor mapa topológico do ambiente medida pela divergência de KL.

5 Conclusões

Devido à segmentação das informações sensoriais, a arquitetura de MEMs em cascata mostrou-se capaz de gerar mapas mais próximos dos modelos reais para a grande maioria dos experimentos realizados porém, o tempo gasto para encontrar esse modelo é muito maior que para a arquitetura básica de geração de MEMs. Essa relação qualidade/tempo deve ser o fator chave para

Tabela 1: Média das 100 realizações da Divergência de KL para os experimentos de diferentes Modelos de MEMs para T variando entre 100 e 800 passos

HMM em Cascata	KL			
Modelo / T	100	200	400	800
(1 - 2)	1,38	0,88	0,59	0,33
(2,1 - 4,2)	1,12	0,92	0,69	0,45
(2,1 - 8,2)	0,94	0,85	0,79	0,62
(2,1 - 16,2)	1,18	1,08	0,97	1,20
(2,1 - 32,2)	1,27	1,08	1,03	1,16
(4,1 - 8,4)	0,89	0,68	0,65	0,43
(4,1 - 16,4)	1,35	1,17	1,18	1,07
(4,1 - 32,4)	1,30	1,11	1,18	1,08
(8,1 - 16,8)	1,40	1,47	1,32	1,65
(8,1 - 32,8)	1,42	1,47	1,41	1,63
(16,1 - 32,16)	1,48	1,60	1,53	1,73
(4,2,1 - 8,4,2)	1,11	0,89	0,64	0,65
(4,2,1 - 16,4,2)	1,23	0,86	0,95	0,78
(4,2,1 - 32,4,2)	1,11	1,10	0,92	0,98
(8,2,1 - 16,8,2)	1,08	0,97	0,93	0,90
(8,2,1 - 32,8,2)	1,28	1,06	0,86	0,98
(8,4,1 - 16,8,4)	1,20	1,18	1,17	0,99
(8,4,1 - 32,8,4)	1,36	1,22	1,06	1,09
(16,2,1 - 32,16,2)	1,35	1,13	1,02	1,07
(16,4,1 - 32,16,4)	1,39	1,25	1,10	1,01
(16,8,1 - 32,16,8)	1,39	1,01	1,35	1,34
(8,4,2,1 - 16,8,4,2)	1,23	0,89	0,73	0,62
(8,4,2,1 - 32,8,4,2)	1,14	0,87	0,90	0,75
(16,4,2,1 - 32,16,4,2)	1,05	0,98	0,88	0,77
(16,8,2,1 - 32,16,8,2)	1,22	0,91	0,90	0,77
(16,8,4,1 - 32,16,8,4)	1,33	1,10	0,95	0,97
(16,8,4,2,1 - 32,16,8,4,2)	1,02	0,81	0,78	0,64

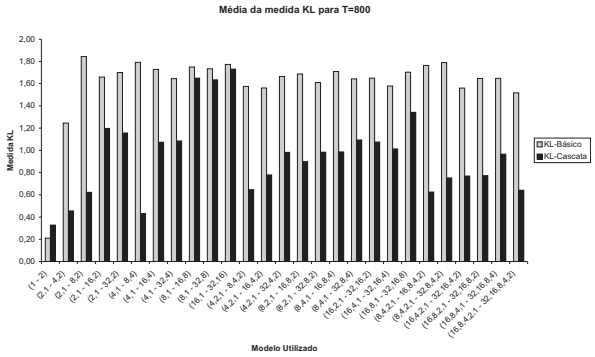

Figura 6: Comparativo entre as arquiteturas testadas avaliadas pela divergência de KL para $T = 800$ passos

escolha da arquitetura em cascata a ser utilizada para a geração de mapas.

É importante ressaltar que, devido ao fato de ser um algoritmo *offline*, a questão tempo não possui uma importância tão grande em relação aos algoritmos que operam em tempo real.

Devido ao fato de que os MEMs geram mapas de estrutura topológica, não é possível efetuar uma comparação de qualidade do mapa gerado em relação aos métodos baseados em estruturas métricas e/ou geométricas como por exemplo, Grades de Ocupação (Elfes, 1987).

É possível que usando um bom algoritmo para inicialização das matrizes envolvidas no treinamento dos MEMs se tenha também melhores resultados tanto em qualidade quanto em tempo de treinamento.

Outra possibilidade que poderá render melhores resultados é a elaboração de uma arquitetura em árvore que, dinamicamente, gera novos sub-modelos em cascata visando tratar de problemas mais complexos, com maior número de estados e de transições. Esta possibilidade seria alcançável através de uma integração entre a arquitetura de MEMs em cascata e a arquitetura hierárquica de MEMs.

Outro fator de interesse é o estudo de uma técnica que, dados valores de N,M,V e T do modelo, informe a melhor arquitetura em cascata (ou não) que cumpra requisitos de tempo e qualidade do mapa a ser gerado.

Agradecimentos

À CAPES pelo suporte financeiro.

Referências

Baum, L. E., Petrie, T., Soules, G. and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains, *The Annals of Mathematical Statistics* **41**(1): 164–171.

Britto, Alceu de Souza, e. a. (2001). Técnicas em processamento e análise de documentos manuscritos, *RITA* **8**(2).

Elfes, A. (1987). Sonar-based real-world mapping and navigation, *IEEE Journal of Robotics and Automation* **RA-3**(3).

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency, *Annals of Mathematical Statistics* **22**: 79–86.

Marceau, G., Ladd, A. M., Bekris, K. E., Rudys, A., Kavraki, L. E. and Wallach, D. S. (2002). Robotics-based location sensing using wireless ethernet, *Mobicom* .

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition, *IEEE - Proceedings* **77**(2).

Shatkay, H. (1998). Learning models for robot navigation, *Technical Report CS-98-11*. http://citeseer.ist.psu.edu/shatkay98learning.html.

Shatkay, H. and Kaelbling, L. P. (1997). Learning topological maps with weak local odometric information, *IJCAI (2)*, pp. 920–929. http://citeseer.ist.psu.edu/shatkay97learning.html.

Wagner, E. R. and Ribeiro, C. H. C. (2006). Cascateamento de modelos escondidos de markov para geração de mapas topológicos em robótica móvel, *XVI Congresso Brasileiro de Automática* .