

Uma Proposta de Mapeamento do Ambiente Exógeno de Sistemas Multi-Agentes Usando Visão Computacional

Douglas Bernardino¹, Leandro Botelho², Carlos Eduardo Pantoja¹

¹Centro Federal de Educação Tecnológica (CEFET/RJ)
– Rio de Janeiro – RJ – Brasil

²Blueshift Brasil
– São Paulo — SP – Brasil

{douglas.bernardiino, leandrobotelhoalves}@gmail.com

pantoja@cefet-rj.br

Abstract. *The dumping of garbage in the seas, which millions of tons of garbage reach the oceans every year, harms the environment and can even lead to the death of species of aquatic living beings. One proposal to mitigate this problem is autonomous boats that scan a certain part of the sea and collect solid waste that floats on the surface. Existing proposals allow the use of guided boats or completely autonomously. However, current approaches do not consider the performance of cognitive agents and a mixed approach to autonomy, which can reduce search costs in certain regions. The objective of this work is to propose a Multi-Agent System focused on not only mapping but also monitoring an exogenous environment through a camera, using Machine Learning and Deep Learning algorithms for object identification and collection to a Multi-agent System.*

Resumo. *O despejo de lixo nos mares, o qual milhões de toneladas de lixo chegam aos oceanos todos os anos, prejudica o meio ambiente e até pode levar a morte de espécies de seres vivos aquáticos. Uma proposta para mitigar esse problema são barcos autônomos que façam a varredura de certa parte do mar e a coleta de resíduos sólidos que flutuam na superfície. As propostas existentes permitem a utilização de barcos guiados ou de forma totalmente autônoma. Contudo, as atuais abordagens não consideram a atuação de agentes cognitivos e uma abordagem mista de autonomia, as quais podem reduzir custos de buscas em determinadas regiões. O objetivo deste trabalho é propor um modelo de visão computacional capaz de mapear um ambiente exógeno através de uma câmera, utilizando algoritmos de Machine Learning e Deep Learning, para identificação de objetos e protótipos informando seus posicionamentos para um Sistema Multi-Agentes Embarcado.*

1. Introdução

Em um relatório publicado em 2021, a Organização das Nações Unidas enfatiza que a poluição aumentou significativamente ao longo dos anos e deve dobrar até 2030, afetando o estilo de vida humana, a biodiversidade e o clima. O monitoramento e recolhimento de resíduos sólidos são tarefas as quais dependem de intervenção humana

para condução de embarcações especializadas em coleta e na identificação de resíduos. Um Sistema Multi-agente (SMA) é composto por diversos agentes cognitivos que podem atuar de forma autônoma e pró-ativa em ambiente físico para atingir um objetivo em comum [Wooldridge 2000]. Este tipo de sistemas é um candidato a reduzir a dependência humana e na identificação de resíduos, dado suas características.

Existem trabalhos que utilizam SMA para monitorar ambientes aquáticos, como o controle de acesso de barcos autorizados em uma marina na região norte da Itália [Armano and Vargiu 2009], e pela iniciativa de uma *start-up* de Hong Kong que desenvolveu uma embarcação equipada com IA para reconhecer e registrar os tipos de lixo coletados e sua localização [Riccio 2022]. Em contraste com os trabalhos anteriores, esta proposta envolve um barco autônomo com um SMA Embarcado [Brandão et al. 2021] que realiza a coleta autônoma dos resíduos presentes na superfície da água. Para alcançar esse objetivo, é essencial contar com um modelo que possa identificar tanto a posição do barco quanto dos resíduos e transmitir essas informações ao SMA. Dessa forma, o barco autônomo se tornaria uma solução viável para enfrentar o desafio da coleta de resíduos em uma baía, permitindo uma atuação eficiente e consciente do meio ambiente. Com o uso do modelo cognitivo *Belief-Desire-Intention* (BDI) [Bratman 1987], os agentes seriam capazes de raciocinar baseados em crenças, desejos e intenções, podendo melhorar a tomada de decisões e a coordenação das atividades de coleta.

Nesse artigo, será proposto um modelo de visão computacional que seja capaz de identificar protótipos de um veículo terrestre, um objeto específico (uma bola) e montar automaticamente as crenças para os agentes. Uma câmera específica será posicionada no topo de uma sala para identificar os objetos e protótipos que estiverem no quadro da câmera. Uma vez identificados, o modelo encaminhará para os agentes as percepções do protótipo e dos objetos para que este possa deliberar. Para isso serão utilizados a linguagem Python para treinar uma máquina usando *Machine Learning*, junto com um dispositivo Kinect para mapeamento e captura de informações. O SMA embarcado será programado utilizando o framework JaCaMo [Boissier et al. 2013] com agentes especializados em interface de hardware [Pantoja et al. 2016] e agentes que se comunicam com a IoT [Pantoja et al. 2018]. Para identificação de objetos será utilizada a biblioteca Open Source Computer Vision Library (OpenCV) que possui módulos de Processamento de Imagens, estrutura de dados, álgebra linear e algoritmos de Visão computacional como filtros de imagem, calibração de câmera, reconhecimento de objetos, análise estrutural e outros, tendo seu processamento em tempo real de imagens, apoiada pelo framework TensorFlow que é uma biblioteca criada para aprendizado de máquina, computação numérica entre outros, tornando-se uma das principais ferramentas para Machine e Deep Learning.

As informações da localização dos resíduos serão encaminhadas a embarcação a partir de uma rede da Internet das Coisas (*Internet of Things* — IoT) usando o Context-Net [Endler et al. 2011] ou através de uma Interface de Programação de Aplicação — *Application Programming Interface* (API). Espera-se que com uma arquitetura híbrida apoiada pela IoT e a API com monitoramento externo ao SMA embarcado, o modelo informe o posicionamento dos objetos e protótipos adequadamente. Para validar a proposta deste trabalho, optou-se por escolher um modelo terrestre para validar a proposta por facilidade na montagem do *setup* de testes. Esta proposta divide-se em: na Seção 2 são apresentados os conceitos básicos; na Seção 3 a arquitetura do projeto e direcionamentos

são apresentados. Por fim, a Seção 4 apresentam algumas considerações finais.

2. Conceitos Básicos

A tecnologia de agentes surge como uma alternativa para apoiar uma resolução cooperativa de sistemas distribuídos, apresentando vantagens em relação as arquiteturas monolíticas tradicionais. Os SMA são sistemas baseados em agentes os quais grupos de agentes trabalham juntos como um único sistema para cooperarem ou competir por seus objetivos [Russel and Norvig 2004]. A visão computacional é a área da ciência que estuda métodos de aquisição, processamento, análise e entendimento de imagens e, no geral, informações do mundo real, com o intuito de produzir dados numéricos ou simbólicos para realizar alguma função ou tomar decisões [Klette 2014]. Neste trabalho, será necessário treinar um modelo de visão computacional utilizando a linguagem Python e bibliotecas como o OpenCV [Harvey 2010] e TensorFlow. Esse modelo será capaz de identificar o protótipo de um veículo terrestre e uma bola nas imagens capturadas pelo dispositivo Kinect, fornecendo assim as informações necessárias para os agentes do SMA embarcado.

3. Sistema Cognitivo de Monitoramento Aquático

Nesta seção serão apresentadas a metodologia e a arquitetura do sistema cognitivo utilizando o framework JaCaMo para monitoramento de resíduos sólidos superficiais em água. Por exemplo, na Baía de Guanabara, no Rio de Janeiro, frequentemente encontram-se resíduos provenientes do descarte inapropriado de diversos tipos de lixo (Figura 1). De acordo com uma matéria publicada pelo jornal O Globo, a concentração de resíduos na baía de Guanabara é de 100 toneladas por dia [Carvalho 2022]. Sistemas cognitivos podem auxiliar em prover pró-atividade e autonomia de para identificar e coletar resíduos.



Figura 1. Foto da Baía de Guanabara

Neste cenário, um barco com um SMA embarcado será capaz de se locomover pelas águas e recolher de forma autônoma os resíduos sólidos que estiverem flutuando. Este barco receberá através de uma rede da IoT as localizações de onde existem resíduos, que são identificados automaticamente por uma câmera monitorando 24h a baía. O Barco utilizará um SMA utilizando o JaCaMo com arquiteturas customizadas de agentes Argo,

o qual é responsável pelo interfaceamento com o Hardware e agentes Comunicadores, que se comunicam com um servidor IoT usando o ContextNet. O JaCaMo foi escolhido por se tratar de um *framework* com ampla adoção na academia e contar com as extensões para hardware e comunicação. A Figura 2 exibe a arquitetura do sistema de monitoramento.

Para a identificação dos resíduos sólidos em água, uma câmera seria responsável por monitorar constantemente certa região da Baía de Guanabara. A partir desse monitoramento, serão armazenadas imagens que auxiliarão a criar uma base de dados supervisionada para identificar quando há ou não resíduos. A partir daí, algoritmos de Aprendizado de Máquina (*Machine Learning* — ML) poderão ser utilizados para mapear em quadrantes as regiões da Baía. Uma vez identificadas regiões com potenciais resíduos sólidos, o sistema de monitoramento enviará uma notificação ao barco utilizando a rede IoT ou uma API. O agente Comunicador do SMA embarcado no barco receberá a notificação e o agente Argo de forma autônoma irá obter seu posicionamento atual através do Sistema Global de Localização (*Global Positioning System* — GPS). Essas informações serão utilizadas para que o SMA identifique em qual quadrante ele está localizado e para qual quadrante ele precisa se deslocar.

Levando em consideração a dificuldade inicial em reproduzir o cenário descrito, foi substituído o barco autônomo por veículo terrestre autônomo e o lixo por uma bola de plástico. De forma que um Kinect ficará disposto a aproximadamente 2 metros do chão, o carro irá ficar no chão na posição inicial (1,1) e a bola será colocada em vários pontos diferentes no chão, levando em consideração os limites dos quadrantes baseado na percepção da câmera. Desta forma, um veículo poderá realizar a coleta de uma bola, em um ambiente exógeno utilizando as informações recebidas de uma rede IOT ou de uma API em tempo real através de uma solução atrelada a câmera de monitoramento (Kinect). O veículo será capaz de se locomover por quadrantes associados a informação recebida pelo agente, sendo o carro acionado somente se existe identificação do objeto bola.

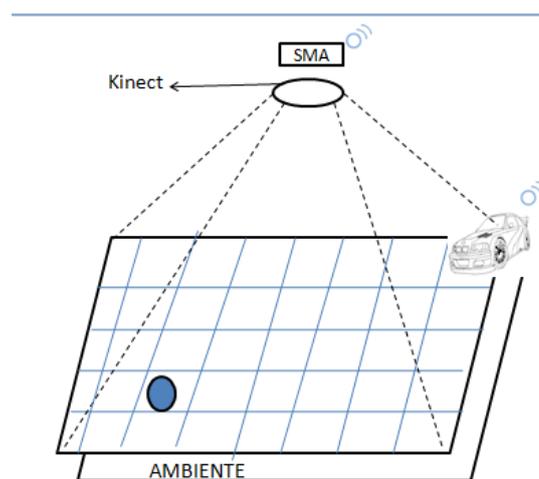


Figura 2. Modelo Conceitual Inicial

3.1. Primeiras Impressões

Inicialmente, a nível de implementação, foram definidas algumas construções a nível de SMA como as crenças, intenções e planos do agente que controlará o barco e procurará

o lixo, chamados respectivamente de **boat** e **garbagefinder**. O **boat** possui uma posição inicial com coordenadas, por exemplo, números inteiros positivos como uma crença. A única intenção inicial é a *hasGarbage* a qual realizará a movimentação nos eixos abscissas e ordenadas no caso de existir uma crença da posição do lixo e do barco. Os planos *hasGarbage* e *move* consiste inicialmente em verificar se o **boat** está em cima do resíduo, se a resposta for positiva, o mesmo irá coleta-lo. O plano *move* é responsável por mover o **boat** até que ele chegue na posição onde foi identificado o lixo.

Primeiramente, o barco é iniciado em uma posição (1,1). Após o recebimento das posições do barco e lixo, acionará plano *hasGarbage* o qual checará se a posição do barco é diferente da posição do lixo. Caso as posições sejam diferentes mover-se-á inicialmente de forma unitária, ora positiva ora negativa, no eixo das abscissas até que sejam iguais. Igualando-se, o agente **boat** movimentar-se-á, agora, no eixo das ordenadas, novamente, de forma unitária ou positivamente ou negativamente, como fosse traçar uma plano cartesiana até que alcance o lixo. Ademais a coleta, o agente **garbagefinder** mandará mensagem avisando ao outro dispositivo que foi realizada a coleta com sucesso. A Figura 3 mostra a implementação do agente **boat** e o *output* da execução do SMA.

The screenshot shows an IDE with a project named 'boat.mas2'. The code for 'boat.asl' is as follows:

```

1 position(1,1).
2
3 !hasGarbage.
4
5 +=hasGarbage: garbage(X,Y) & position(Z,W) & ((X \== Z) | (Y \== W)) <-
6   !print;
7   !move.
8
9 +=!move: garbage(X,Y) & position (Z,W) & X > Z <-
10  --position(Z+1,W);
11  !hasGarbage.
12
13 +=!move: garbage(X,Y) & position (Z,W) & X < Z <-
14  --position(Z-1,W);
15  !hasGarbage.
16
17 +=!move: garbage(X,Y) & position (Z,W) & X == Z & Y > W <-
18  --position(Z,W+1);
19  !hasGarbage.
20
21 +=!move: garbage(X,Y) & position (Z,W) & X == Z & Y < W <-
22  --position(Z,W-1);
23  !hasGarbage.
24
25 +=!print: position(Z,W) <-
26  .print("Xbarco: ",Z," e Ybarco:",W).
27
28 +=hasGarbage: garbage(X,Y) & position(Z,W) & (X == Z) & (Y == W) <-
29  .print("Coletou em ",Z,"x",W,"");
30  -garbage(X,Y){source(finder)};
31  !hasGarbage.
32
33

```

The console output shows the execution of the agent, including the initial position and the collection of garbage:

```

MAS Console - boat
[boat] Xbarco:1 e Ybarco:1
[boat] Xbarco:2 e Ybarco:1
[boat] Xbarco:3 e Ybarco:1
[boat] Xbarco:5 e Ybarco:1
[boat] Xbarco:5 e Ybarco:2
[boat] Coletou em:5.
[boat] Xbarco:5 e Ybarco:3
[boat] Xbarco:4 e Ybarco:3
[boat] Xbarco:3 e Ybarco:3
[boat] Xbarco:2 e Ybarco:3
[boat] Xbarco:2 e Ybarco:4
[boat] Xbarco:2 e Ybarco:4
[boat] Coletou em:2.5.

```

Figura 3. A programação do agente boat.

Na versão atual, algumas limitações ainda existem, tais como: o agente **boat** só consegue lidar com uma informação de lixo por vez e caso receba uma outra informação de outro lixo, poderá ocorrer um desvio de percurso. Além disso, existe ausência na verificação se há um outro dispositivo próximo, podendo haver colisões.

4. Discussão

Por ser um trabalho em andamento, o projeto apresenta as tecnologias iniciais e o funcionamento do SMA de forma que possa no futuro ser integrado ao projeto finalizado. A proposta é utilizar um SMA embarcado e a IoT ou uma API para o monitoramento e detecção de objetos para a coleta destes por um carro autônomo ou por um barco autônomo.

Como trabalhos futuros, é planejado calibrar a câmera Kinect, pesquisar uma forma de comunicação entre o código Python e o framework de programação JaCaMo, realizar o treinamento de máquina, realizar testes em ambientes reais e, posteriormente, escrever sobre os resultados obtidos com o protótipo terrestre e marinho em ambiente

controlado (piscina). Inicialmente, não será levado em consideração a atuação do vento, das correntes marítimas, outros agentes coletando lixo, o cálculo da menor distância entre os pontos do barco e do lixo mais próximo, a interação com outros barcos e outros fatores naturais. Posteriormente, com as fases mencionadas concluídas, as questões acima serão abordadas. Dado a dificuldade em resolver esses problemas em ambiente marinho foi optado por iniciar o projeto considerando um ambiente terrestre.

Referências

- Armano, G. and Vargiu, E. (2009). A multiagent system for monitoring boats in marine reserves. In *International Workshop on Programming Multi-Agent Systems*, pages 254–265, Berlin. Springer.
- Boissier, O., Bordini, R. H., Hübner, J. F., Ricci, A., and Santi, A. (2013). Multi-agent oriented programming with jacamo. *Science of Computer Programming*, 78(6):747–761.
- Brandão, F. C., Lima, M. A. T., Pantoja, C. E., Zahn, J., and Viterbo, J. (2021). Engineering approaches for programming agent-based iot objects using the resource management architecture. *Sensors*, 21(23). <https://doi.org/10.3390/s21238110>.
- Bratman, M. E. (1987). *Intention, Plans and Practical Reasoning*. Cambridge Press.
- Carvalho, Janaina e Lima, M. (2022). Baía de guanabara agoniza com despejo de quase 100 toneladas de lixo por dia, 30 anos após a eco-92. *O Globo*.
- Endler, M., Baptista, G., Silva, L., Vasconcelos, R., Malcher, M., Pantoja, V., Pinheiro, V., and Viterbo, J. (2011). Contextnet: context reasoning and sharing middleware for large-scale pervasive collaboration and social networking. In *Proceedings of the Workshop on Posters and Demos Track*, page 2. ACM.
- Harvey, A. (2010). Opencv face detection: Visualized. *Vimeo [En lúnia] Disponible: <https://vimeo.com/12774628>*.
- Klette, R. (2014). *Concise computer vision*, volume 233. Springer.
- Pantoja, C. E., Soares, H. D., Viterbo, J., and El Fallah-Seghrouchni, A. (2018). An architecture for the development of ambient intelligence systems managed by embedded agents. In *SEKE*, pages 215–220. <https://doi.org/10.18293/SEKE2018-110>.
- Pantoja, C. E., Stabile, M. F., Lizarin, N. M., and Sichman, J. S. (2016). ARGO: An Extended Jason Architecture that Facilitates Embedded Robotic Agents Programming. In Baldoni, M., Müller, J. P., Nunes, I., and Zalila-Wenkstern, R., editors, *Engineering Multi-Agent Systems*, pages 136–155, Cham. Springer International Publishing. https://doi.org/10.1007/978-3-319-50983-9_8.
- Riccio, G. (2022). Clearbot neo, barco robótico que limpa portos e rios com a ajuda da ia. *Futuro próximo*.
- Russel, S. and Norvig, P. (2004). *Inteligência Artificial*.
- Wooldridge, M. J. (2000). *Reasoning about rational agents*. MIT press.