Projeto automático de controladores para robôs de enxame em diferentes condições de cenário

Fernando J. Mendiburu * Marcos R. A. Morais *
Antonio M. N. Lima *

* Programa de Pós-Graduação em Engenharia Elétrica - PPGEE
Departamento de Engenharia Elétrica - DEE
Universidade Federal de Campina Grande - UFCG
Rua Aprígio Veloso, 882, 58429-900 Bairro Universitário
Campina Grande, Paraíba, Brazil
e-mails: fernando.mendiburu@ee.ufcg.edu.br, morais@dee.ufcg.edu.br,
amnlma@dee.ufcg.edu.br.

Abstract: One of the main limitations of the automatic *offline* design methods is the *reality gap* problem. The *reality gap* is defined as the performance drop that is obtained in the evaluation of collective behavior in several environments (ex. real and simulated) by the non modeled characteristics during the synthesis, that are manifested in environments in which the controller was not designed. The reality gap is one of the biggest problems in the automatic design of controllers for swarm robots. In the following article is presented a set of experiments that shows how the representation of the scenario influences on the performance of the controllers. An imprecise representation of the scenario has a negative influence on the automatic *offline* design, leading in the worst case to failure the collective task.

Resumo: Uma das principais limitações dos métodos automáticos *offline* é o *reality gap*. O *reality gap* é definido como as diferenças de desempenho que se obtêm na avaliação do comportamento coletivo em diversos ambientes (real e simulado) pelas características não modeladas durante o projeto de síntese. Estas diferenças de desempenho se manifestam nos ambientes para o qual o controlador não foi projetado. O *reality gap* é um dos maiores problemas no projeto automático de controladores para robôs de enxame. No seguinte artigo são apresentados um conjunto de experimentos que mostram como a representação do cenário de trabalho, influencia no desempenho dos controladores. Uma representação imprecisa do cenário influencia de forma negativa no projeto automático *offline* de controlador, levando, no pior dos casos, ao fracasso da realização da tarefa coletiva.

Keywords: Swarm robotics; Automatic *offline* design of controllers; Representation of scenario conditions.

Palavras-chaves: Robótica de enxame; projeto automático de controladores; representação das condições do ambiente.

1. INTRODUÇÃO

Os enxames de robôs são compostos por uma grande quantidade (dezena, centena ou milhar) de robôs de capacidades limitadas que devem cooperar para realizar tarefas de relevância (Şahin, 2004). Estes sistemas se apresentam promissores em aplicações medicas, industriais e militares. Exemplos dessas aplicações são encontrados na industria de manufatura, monitoramento de produção, monitoramento de fronteiras, reconhecimento de terrenos, exploração espacial, resgate de vítimas em catástrofes, manipulação de matérias perigosos, dentre outras aplicações de relevância para a humanidade (Brambilla et al., 2013).

Estes sistemas de enxame de robôs são compostos por robôs autônomos que compõem um sistema auto-organizado, pois são utilizados controladores descentralizados e as decisões são tomadas por cada robô de forma individual. Por

sua vez, os robôs do enxame operam utilizando unicamente informações locais do ambiente e são sistemas altamente redundantes, pois são compostos normalmente de robôs homogêneos. Estas características impõem a escalabilidade, flexibilidade e tolerância a falhas do sistema que permitem que os robôs resolvam tarefas de fundamental importância na robótica (Brambilla et al., 2013; Garattoni and Birattari, 2016).

De forma geral, no projeto de controlador em enxames de robôs são ajustados a estrutura e parâmetros do controlador individual de cada robô do enxame, para os robôs se locomover e atuar no ambiente de trabalho. O controlador individual do robô é a porção de software computacional que recebe informação de sensores, processa e produz atuação (Nelson et al., 2009). Este controlador é o que realiza o mapeamento dos sensores e atuadores

do robô. A execução deste controlador projetado em cada robô do enxame permite desenvolver um conjunto de tarefas com os outros integrantes do enxame. O projeto de controlador individual de cada robô do enxame pode ser abordado com técnicas manuais, semi-automáticas ou automáticas (Brambilla et al., 2013).

No projeto automático de controlador para enxames de robôs, o projeto deve encontrar o controlador para cada robô individual. No projeto *offline* (Birattari et al., 2019), realizado *a priori* dos robôs ser instanciados no cenário de trabalho, o projeto é realizado por intermédio de sucessivas avaliações do comportamento coletivo do enxame no ambiente em que devem desenvolver a tarefa coletiva (Brambilla et al., 2013). A síntese é guiada por intermédio de um algoritmo de otimização que deve maximizar, ou minimizar, uma função objetivo para decidir entre as possíveis estruturas de controle e ajustar os parâmetros da arquitetura. Assim, nesta abordagem, a tarefa é especificada no nível macroscópico como uma especificação de usuário (a função objetivo), que guia um algoritmo de otimização para a escolha do controlador de melhor desempenho. Este controlador instanciado em cada robô definirá o comportamento coletivo do enxame que resolva essa tarefa.

Uma das principais limitações dos métodos automáticos *offline* é o *reality gap*. O *reality gap* é definido como as diferenças de desempenho que se obtêm na avaliação do comportamento coletivo devidas a mudanças no ambiente (Urzelai and Floreano, 2000). Mudanças no ambiente são devidas a vários fatores, como a representação das condições ambiente (ex. condições de luminosidade), transferência do controlador entre o simulador e o mundo físico e a transferência de controlador entre diferentes plataformas robóticas (Urzelai and Floreano, 2000). O *reality gap* se manifesta porque características não representadas durante o projeto de síntese, influenciam nos ambientes para o qual o controlador não foi projetado. O *reality gap* é um dos maiores problemas no projeto automático de controladores para robôs de enxame (Francesca and Birattari, 2016).

Diversos trabalhos estudam o problema do *reality gap* (Ligot and Birattari, 2018), por intermédio da representação fiel das características dos robôs e do ambiente (Jakobi et al., 1995; Koos et al., 2013; Miglino et al., 1995; Zagal and Ruiz-Del-Solar, 2007). Por outro lado, outros trabalhos assumem estas diferenças e se focam em produzir controladores que sejam robustos a estas características não modeladas (Francesca et al., 2014, 2015; Urzelai and Floreano, 2000; Watson et al., 2002; Duarte et al., 2015).

As diferenças de desempenho entre simulação (onde o controlador foi projetado) e o cenário real (onde o controlador é de fato executado) são produzidas pelas características não modeladas dos robôs e das características não modeladas do ambiente. Estas diferenças fazem com que o controlador projetado não consiga resolver corretamente a tarefa coletiva quando instanciado no cenário real. Em certas circunstâncias, algumas das características do ambiente são desconhecidas *a priori* ou conhecidas parcialmente.

Por exemplo, suponha um cenário de catástrofe em que robôs devem operar para resgate e salvamento de vítimas. As condições do cenário *a priori* não são conhecidas totalmente, tais como o número de pessoas envolvidas, as

condições do terreno, a quantidade e posição de obstáculos dentro do cenário, as condições de luminosidade, dentre outras características. Métodos de projeto *offline* são susceptíveis a este tipo de incertezas: divergências no que diz respeito as condições do cenário quando é realizada a representação do ambiente, podem fazer com que os controladores produzidos nessas condições possuam dificuldades para resolver de fato a tarefa no cenário real de trabalho. Como o método de projeto *offline* sintetiza os controladores avaliando o desempenho do enxame em simulação, o controlador individual projetado é ajustado a esse modelo específico (Ligot and Birattari, 2018), o qual não reproduz perfeitamente o cenário real de trabalho (Floreano et al., 2008).

Portanto, a informação parcial ou incompleta das condições do ambiente no qual os robôs devem desenvolver a tarefa coletiva influenciam de forma negativa no projeto automático *offline* de controlador, levando, no pior dos casos, ao fracasso da realização da tarefa coletiva. Na seguinte seção são apresentados um conjunto de experimentos que mostram como divergências na representação do cenário de trabalho no que diz respeito ao cenário real, produzem queda significativa do desempenho dos controladores.

2. METODOLOGIA

Nesta seção, é descrito o modelo do robô, controlador produzido pelo método de projeto, protocolo utilizados nos experimentos, a descrição do cenário de trabalho e as missões que os robôs devem realizadas.

2.1 Modelo de Robô

O método AUTOMODE-CHOCOLATE produz controladores para o robô e-puck (Mondada et al., 2009). O e-puck é uma plataforma de robô projetada para fins de pesquisa e educação. É equipado com o computador Overo Gumstick, um sensor que mede a distância relativa a outros robôs (sensor R&B), três sensores infravermelhos de chão, uma câmera omnidirecional e oito sensores infravermelhos que medem a intensidade da luz de proximidade e ambiente.

Os robôs e-puck possuem tração diferencial e são representados pelo modelo cinemático do unicycle da Equação (1).

$$\begin{cases} \dot{x} = v \cos(\theta) \\ \dot{y} = v \sin(\theta) \\ \dot{\theta} = \omega \end{cases} \quad (1)$$

A velocidade $v \in \mathbb{R}$ é a velocidade linear do robô e $\omega \in \mathbb{R}$ a velocidade angular do robô. A pose do robô é representada com as variáveis x , y e θ representada em relação ao sistema de coordenadas fixo no cenário de trabalho.

O modelo de referência do robô e-puck foi apresentado em Francesca et al. (2014). O modelo de referência é uma representação que define as características e capacidades da plataforma robótica, tais como os sensores e atuadores que são disponíveis no controlador em cada ciclo de controle (Francesca et al., 2015). Cada sensor (atuador) é abstraído em variáveis que serão lidas (escritas) pelo controlador para realizar o controle da plataforma. Estes sensores foram modelados como explicitado a seguir. Os sensores de proximidade permitem medir a distância de

entidades próximas ao robô. O valor medido do sensor de proximidade $i \in \mathbb{N} : [1, 8]$ é $v_{i_{prox}} \in \mathbb{R} : [0, 1]$, modelado por intermédio da Equação (2).

$$v_{i_{prox}} = ad_{ij}^2 + bd_{ij} + c + \sigma_{prox}, \quad (2)$$

em que $a = 298,7$; $b = -36,9$ e $c = 1,1$. Estas constantes foram obtidas de dados empíricos do sensor de proximidade ajustados a um modelo de segundo grau. A distância euclidiana d_{ij} (em metros) é entre o sensor $i \in \mathbb{N} : [1, 8]$ e o obstáculo mais próximo do sensor na linha de visão. O valor do sensor é diferente de zero quando $d_{ij} \leq 0,05m$. De acordo com dados experimentais, o ruído é considerado um ruído aleatório uniforme definido no intervalo $\sigma_{prox} \in \mathbb{R} : [-0,05; 0,05]$.

Os sensores de luminosidade permitem medir a presença de luz de uma fonte luminosa. Neste sensor, são consideradas oclusões da fonte luminosa com obstáculos ou robôs. Caso existissem mais de uma fonte luminosa o cálculo é feito somando o valor de cada fonte separadamente sem considerar a interferência entre fontes luminosas. O valor medido do sensor de luminosidade $i \in \mathbb{N} : [1, 8]$ é $v_{i_{luz}} \in \mathbb{R} : [0, 1]$, é modelado por intermédio da Equação (3).

$$v_{i_{luz}} = \left(e^{-\frac{2d_{ij}}{I}} \right) s_{\theta} + \sigma_{luz}, \quad (3)$$

em que d_{ij} é a distância euclidiana entre o sensor $i \in \mathbb{N} : [1, 8]$ e o obstáculo mais próximo do sensor na linha de visão. A distância máxima considerada é $d_{ij} \leq 2,5m$. No simulador, a intensidade da fonte luminosa é medida em metros e foi definida num valor de $I = 2,5m$ considerando a distância máxima de percepção do estímulo luminoso no sensor físico. A ponderação $s_{\theta} \in \mathbb{R} : [0, 1]$ permite ponderar linearmente caso os sensores não estejam diretamente posicionados na fonte luminosa. Por exemplo, um sensor diretamente posicionado em direção à fonte luminosa ($\theta_{dif} = 0$, ou seja o sensor aponta diretamente) possui um valor de $s_{\theta} = 1$ e um sensor a mais de $\theta_{dif} \geq \pi/2$ da fonte luminosa possui um valor $s_{\theta} = 0$ (a fonte luminosa não é observada). De acordo com dados experimentais, o ruído é considerado como aleatório uniforme definido no intervalo $\sigma_{luz} \in \mathbb{R} : [-0,05; 0,05]$.

O sensor de proximidade relativa ($R\&B$) permite realizar a medida de ângulo e distância relativa com robôs da vizinhança. O sensor é modelado por intermédio da Equação (4).

$$v_{i_{R\&B}} = d_{ij} + \sigma_{R\&B}. \quad (4)$$

em que d_{ij} é a distância euclidiana entre o robô focal $i \in \mathbb{N} : [1, N]$ e o robô $j \in \mathbb{N} : [1, N]$, $i \neq j$, de um conjunto de N robôs. De acordo com dados experimentais, a modelagem considera ruído aleatório gaussiano com média nula e desvio padrão $\sigma_{R\&B} \in \mathbb{R} : [-0,1; 0,1]m$. O sensor de chão foi considerado um sensor ideal para a detecção de três valores possíveis: chão preto, cinza e branco. A distância máxima de detecção é de $d_{ij} \leq 20cm$.

Com relação aos atuadores do robô, foi considerado um modelo que possui uma flutuação aleatória em cada uma das velocidades lineares das rodas do robô. Na Equação (5) observa-se o modelo para ambos atuadores do robô, com v_r e v_l a velocidade linear da roda direita e esquerda, respectivamente.

Por sua vez, v'_r e v'_l são os comandos de velocidade linear da roda direita e esquerda, respectivamente.

$$\begin{cases} v_r = v'_r + v'_r \sigma_{atu} \\ v_l = v'_l + v'_l \sigma_{atu} \end{cases} \quad (5)$$

Segundo dados experimentais, foi considerado um ruído aleatório uniforme definido no intervalo $\sigma_{atu} \in \mathbb{R} : [-0,05; 0,05]$.

2.2 Controlador

O projeto de controlador é realizado com um método automático modular *offline*, denominado de AUTOMODE-CHOCOLATE (Francesca et al., 2014, 2015). O resultado da síntese é um controlador modular que instanciado em cada robô permite resolver tarefas coletivas definidas por um usuário. Esse controlador modular é uma máquina de estado finita probabilística (Rabin, 1963), composta por módulos de comportamento, módulos de transições, cada um com os seus parâmetros específicos. Os valores mínimo e máximo de cada parâmetro do controlador, assim como também os comportamentos e transições foram definidos da mesma forma que em (Francesca et al., 2014). A busca do controlador é realizada utilizando o algoritmo de otimização F-Race (Birattari et al., 2002) implementado na sua forma iterada (López-Ibáñez et al., 2016). O algoritmo explora o espaço de busca do controlador e decide qual é o controlador de melhor desempenho que executado nos robôs permite resolver a tarefa, eliminando controladores de baixo desempenho. O algoritmo é guiado pela definição de uma função objetivo que descreve a tarefa no nível de enxame e permite catalogar quantitativamente o desempenho de um controlador particular.

2.3 Descrição do cenário de trabalho e missões

Nos seguintes experimentos são apresentados duas tarefas coletivas anteriormente apresentadas na bibliográfica nos artigos (Francesca et al., 2014, 2015). Estas missões apresentam tarefas que são consideradas referência na robótica de enxame. Cada missão descreve por intermédio de uma função objetivo uma tarefa coletiva que os robôs do enxame devem realizar num determinado cenário de trabalho. A Figura 1 e Figura 2 apresentam os cenários com a pose inicial dos robôs numa instância particular de cada experimento. As missões são: *procura de objetos* (Francesca et al., 2014) e *acesso restrito ao refúgio* (Francesca et al., 2015).

Os robôs operam num cenário dodecagonal de $4.91m^2$. O cenário, chamado de arena, é restrito por paredes que limitam a extensão da mesma. Algumas das regiões do chão da arena são sinalizadas pela sua cor, dependendo da missão. É definido um sistema de coordenadas com a origem no centro da arena, unicamente para fins de descrição da arena. Os eixos x^+ e y^+ apontam para acima e para a esquerda, respectivamente.

Procura de objetos: A missão tem o objetivo de realizar agrupamento de objetos encontrados em regiões específicas da arena. Os robôs devem coletar a maior quantidade de objetos possíveis de duas regiões circulares pretas de diâmetro $0,3m$ localizadas em $(-0,75m; 0,0m)$ e $(0,75m; 0,0m)$ para logo depositar esses objetos numa

região branca em $y \leq -0,6m$ e rodeada das paredes da arena. Como os robôs não possuem efetadores para coletar os objetos, se considera que um robô coleta o objeto quando ingressa nas regiões pretas. Por sua vez, é considerado que o robô deposita o objeto na região branca da arena, quando ingressa nela e se antes ingressou na região preta.

Figura 1. Conjunto de cenários da missão *procura de objetos*. Os controladores foram projetados no cenário da esquerda (cenário CPR). Logo esses controladores são avaliados em cenários sem fonte luminosa, no cenário do centro da figura (cenário CSL). Finalmente, os controladores são avaliados num cenário em que a posição da fonte luminosa é modificada, no cenário da figura da direita (cenário CLO). As imagens também apresentam a posição inicial dos robôs.

Na Figura 1 (esquerda) observa-se o cenário para o qual os controladores foram projetados: a arena possui uma fonte luminosa próxima da área onde os objetos são depositados localizada em $(0,0m; -1,5m)$ e a $0,4m$ de altura. Na Figura 1 observa-se, também, duas instâncias da mesma classe de tarefas, nas quais a fonte luminosa possui diferente intensidade (Figura 1 centro) e localização (Figura 1 direita) em relação ao cenário para o qual os controladores foram projetados. Estes dois últimos cenários são considerados os *cenários reais*, onde os robôs devem desenvolver, de fato, a tarefa coletiva.

A função objetivo que deve ser maximizada é

$$C_1(T_m) = n_o(T_m), \quad (6)$$

em que $n_o(T_m)$ é a quantidade de objetos que são depositados corretamente na região branca no final do experimento no instante de tempo T_m . A função objetivo que deve ser maximizada é a quantidade de objetos depositados na região branca da arena. Assim, quanto maior o valor, maior a quantidade de objetos coletados nessa região.

Acesso restrito ao refúgio: A missão tem o objetivo de agrupar a maior quantidade de robôs no refúgio que possui chão branco. Por sua vez, o cenário também possui uma fonte luminosa que serve como orientação dos robôs, da mesma forma que duas regiões circulares pretas de $0,6m$ de diâmetro localizadas em $(-0,8m; 0,1m)$ e $(0,8m; 0,1m)$. A lâmpada está localizada em $(0,0m; -1,5m)$ e a $0,4m$ de altura.

Na Figura 2 (esquerda) observa-se o cenário para o qual os controladores foram projetados: O refúgio tem as dimensões de $0,15m \times 0,6m$ e possui um único acesso livre com os outros três acessos bloqueados por paredes, localizado na origem de coordenadas. Na Figura 2 observa-se, também, duas instâncias da mesma classe de tarefas, nas quais o refúgio possui uma diferença de localização de $y = 0,5m$ (Figura 2 centro) e um cenário em que uma

Figura 2. Conjunto de cenários da missão *acesso restrito ao refúgio*. Os controladores foram projetados no cenário da esquerda (cenário CPR). Logo esses controladores são avaliados em cenários em que a posição do refúgio é modificada, no cenário do centro da figura (cenário CDE). Finalmente, os controladores são avaliados num cenário em que a parede do refúgio foi removida, no cenário da figura da direita (cenário CSP). As imagens também apresentam a posição inicial dos robôs.

parede do refúgio foi removida (Figura 2 direita). Estes dois últimos cenários são considerados os *cenários reais*, onde os robôs devem desenvolver, de fato, a tarefa coletiva.

A função objetivo que deve ser maximizada é

$$C_2(t) = \sum_{t=0}^{T_m} N(t), \quad (7)$$

em que $N(t)$ é a quantidade de robôs no refúgio no instante de tempo t . A função objetivo é avaliada a cada $0,1s$. A função objetivo mede a presença dos robôs no refúgio branco. Assim, quanto maior o valor, maior a presença de robôs na região.

2.4 Protocolo

É comparado AUTOMODE-CHOCOLATE em duas missões sobre diferentes condições de cenário. AUTOMODE-CHOCOLATE é implementado sem modificações como foi definido em (Francesca et al., 2014, 2015). O enxame de robôs é caracterizado por 20 robôs e-puck e o tempo disponível para realizar a missão são 120 s. Cada método de projeto é executado 30 vezes em cada uma das duas missões. O projeto foi realizado sobre os cenários CPR da Figura 1 (esquerda) (Figura 2 (esquerda)), obtendo 30 controladores para cada missão, definidos como máquinas de estado probabilísticas finitas (Francesca et al., 2014, 2015). Logo, é mensurado o desempenho de cada um desses controladores uma única vez para em cada um dos cenários da Figura 1 (Figura 2), respectivamente. Para o projeto de cada controlador foram utilizados 50.000 simulações. Todas as simulações, deste artigo foram realizadas utilizando o simulador ARGOS3, versão 48 (Pinciroli et al., 2012).

São utilizados diagrama de caixas para representar os resultados. Cada amostra do diagrama de caixas representa o desempenho de um controlador avaliado no cenário. Em cada missão os robôs foram posicionados aleatoriamente no início de cada experimento. Ainda, são realizados testes estatísticos utilizando o teste estatístico pareado de Wilcoxon com 95% de significância estatística (Conover, 1999) para realizar a comparações dos desempenhos dos controladores nas diferentes condições do cenário.

Das missões é esperado que o método de projeto consiga abordar a tarefa coletiva no cenário para o qual o contro-

lador foi projetado, nas condições modeladas do cenário. Porém, é esperado observar uma queda de desempenho quando esses mesmos controladores sejam instanciados em cenários que são pequenas variações do cenário para o qual os controladores foram projetados.

3. RESULTADOS

Na Figura 3 observam-se os resultados da missão *procura de objetos*. De acordo ao teste de Wilcoxon, os controladores projetados no cenário CPR (no cenário para o qual foram projetados os controladores) possuem melhor desempenho que esses mesmos controladores em outros cenários da mesma classe de tarefas (nos cenários CSL e CLO). A máxima mediana é produzida no cenário no qual o controlador foi projetado, obtendo um valor de aproximadamente 22 objetos coletados. Comparando as medianas, a queda de desempenho é de aproximadamente 3 objetos coletados (em CSL) e 6 objetos coletados (em CLO). Assim, os resultados mostram a relação do desempenho com as condições de luminosidade. A diferença nas condições de luminosidade prejudicam o desempenho e a solução da tarefa coletiva. Pode ser observado que a ausência da fonte luminosa no cenário CSL faz com que controladores se comportem equivalentemente à solução que realiza exploração aleatória. Por sua vez, a localização da fonte luminosa no cenário CLO produz que os robôs coletem os objetos das regiões pretas e logo se orientem numa direção oposta à direção do depósito dos objetos, prejudicando o desempenho da tarefa coletiva. Pode ser observado que o desempenho dos controladores avaliados em CSL e CLO possuem desempenhos equivalentes à solução CAL, que executa os controladores unicamente com exploração aleatória.

Figura 3. Desempenho dos controladores na missão *procura de objetos* no cenário CPR (os controladores são avaliados no cenário que foram projetados), no cenário CSL e no cenário CLO. Por sua vez, o cenário CAL é similar ao CPR, mas os controladores avaliados realizam unicamente exploração aleatória. A avaliação dos controladores no cenário CAL foi realizada para fins de comparação.

Na Figura 4 observam-se os resultados da missão *acesso restrito ao refúgio*. De acordo ao teste de Wilcoxon, os controladores projetados no cenário CPR (no cenário para o qual foram projetados os controladores) possuem melhor desempenho que esses mesmos controladores em outros cenários da mesma classe de tarefas (nos cenários CDE e CSP). A máxima mediana é produzida no cenário no qual o controlador foi projetado, próxima de 5000 presenças de robôs no refúgio. Comparando as medianas, a queda de desempenho é de aproximadamente 60% (em CDE) e quase 100% (em CSP) na quantidade de presenças de robôs. Assim, os resultados mostram a relação do desempenho com a posição do refúgio. Um deslocamento do refúgio onde os robôs devem se reunir, prejudica a resolução da tarefa coletiva: os robôs possuem dificuldades para se agrupar no refúgio. Por sua vez, a remoção de uma das paredes do refúgio prejudica ainda mais o desempenho: os robôs não conseguem se agrupar no refúgio nem permanecer nele. Os resultados mostram que os controladores projetados em CPR não utilizam a informação do chão branco para se deter e se agrupar no refúgio. É utilizado unicamente a informação da fonte luminosa e informações do perímetro do refúgio. Assim, a remoção de uma das paredes do refúgio, como é feito em CSP, prejudica em grande forma o desempenho dos controladores. Os controladores avaliados em CSP possuem desempenhos equivalentes à solução CAL, que executa os controladores unicamente com exploração aleatória.

Figura 4. Desempenho dos controladores na missão *acesso restrito ao refúgio* no cenário CPR (os controladores são avaliados no cenário que foram projetados), no cenário CDE e no cenário CSP. Por sua vez, o cenário CAL é similar ao CPR, mas os controladores avaliados realizam unicamente exploração aleatória. A avaliação dos controladores no cenário CAL foi realizada para fins de comparação.

4. CONCLUSÃO

No presente artigo foram estudadas as limitações dos métodos automáticos, no que diz respeito ao *reality gap*. Foram estudadas, especificamente, como controladores projetados automaticamente para condições de cenário especifi-

cas, possuem queda significativa do desempenho na resolução da tarefa coletiva quando esses mesmos controladores são avaliados em outros cenários da mesma classe, que possuem variações nas condições do cenário. Foram realizados dois experimentos, um experimento em que os robôs devem realizar coleta de objetos e outro experimento em que os robôs devem se agrupar num refúgio. Foi realizada a representação do cenário de trabalho e foi realizado o projeto de controlador baixo essas condições de cenário. Logo, esses controladores foram avaliados em diferentes condições de cenário, modificando a posição da fonte luminosa, intensidade, modificação do perímetro e modificação da posição de agrupamento dos robôs. Dos resultados observa-se que existe queda de desempenho no comportamento coletivo quando os controladores são avaliados em outras condições de cenário, devido a que o controlador individual projetado é ajustado a uma representação do ambiente específica que pode não generalizar a outras condições de ambiente. Isto mostra que os métodos de projeto automático são sensíveis as condições do ambiente de trabalho. Uma solução possível para que o projeto *offline* seja menos sensível à representação do ambiente é a combinação com técnicas de projeto automático *online* de controladores, que permite o ajuste do controlador projetado *offline*, permitindo a adaptação a diversas condições do ambiente.

AGRADECIMENTOS

Os autores agradecem o apoio financeiro da CAPES e CNPq.

REFERÊNCIAS

- Birattari, M., Stützle, T., Paquete, L., and Varrentrapp, K. (2002). A racing algorithm for configuring metaheuristics. In W. Langdon and et al. (eds.), *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO*, 11–18. Morgan Kaufmann, San Francisco CA.
- Birattari, M., Ligot, A., Bozhinoski, D., Brambilla, M., Francesca, G., Garattoni, L., Garzon Ramos, D., Hasselmann, K., Kegeleirs, M., Kuckling, J., Roli, A., Salman, M., and Stützle, T. (2019). Automatic Off-line Design of Robot Swarms: A Manifesto. *Frontiers in Robotics and AI*, (29), 1–9.
- Brambilla, M., Ferrante, E., Birattari, M., and Dorigo, M. (2013). Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1), 1–41.
- Conover, W.J. (1999). *Practical Nonparametric Statistics*. John Wiley & Sons, New York, third edition.
- Duarte, M., Oliveira, S.M., and Christensen, A.L. (2015). Evolution of Hybrid Robotic Controllers for Complex Tasks. *Journal of Intelligent & Robotic Systems*, 78(3), 463–484.
- Floreano, D., Husbands, P., and Nolfi, S. (2008). Evolutionary robotics. *Handbook of Robotics*, 1423–1451.
- Francesca, G. and Birattari, M. (2016). Automatic design of robot swarms: achievements and challenges. *Frontiers in Robotics and AI*, 3(29), 1–9.
- Francesca, G., Brambilla, M., Brutschy, A., Garattoni, L., Miletitch, R., Podevijn, G., Reina, A., Soleymani, T., Salvaro, M., Pinciroli, C., and Birattari, M. (2015). AutoMoDe-Chocolate: automatic design of control software for robot swarms. *Swarm Intelligence*, 9(2/3), 125–152.
- Francesca, G., Brambilla, M., Brutschy, A., Trianni, V., and Birattari, M. (2014). AutoMoDe: A novel approach to the automatic design of control software for robot swarms. *Swarm Intelligence*, 8(2), 89–112.
- Garattoni, L. and Birattari, M. (2016). Swarm robotics. In J. Webster (ed.), *Wiley Encyclopedia of Electrical and Electronics Engineering*. John Wiley & Sons, Hoboken NJ.
- Jakobi, N., Husbands, P., and Harvey, I. (1995). Noise and the reality gap: the use of simulation in evolutionary robotics. In F. Morán and et al. (eds.), *Advances in Artificial Life*, volume 929 of *LNCS*, 704–720. Springer, London, UK.
- Koos, S., Mouret, J.B., and Doncieux, S. (2013). The transferability approach: crossing the reality gap in evolutionary robotics. *IEEE Transactions on Evolutionary Computation*, 17(1), 122–145.
- Ligot, A. and Birattari, M. (2018). On mimicking the effects of the reality gap with simulation-only experiments. In M.D. et al. (ed.), *Swarm Intelligence, ANTS*, volume 11172 of *LNCS*, in press. Springer, Berlin, Germany.
- López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M., and Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3, 43–58.
- Miglino, O., Lund, H., and Nolfi, S. (1995). Evolving mobile robots in simulated and real environments. *Artificial life*, 2(4), 417–434.
- Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klapotcz, A., Magnenat, S., Zufferey, J.C., Floreano, D., and Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. In P. Gonçalves, P. Torres, and C. Alves (eds.), *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, 59–65. Instituto Politécnico de Castelo Branco, Portugal.
- Nelson, A.L., Barlow, G.J., and Doitsidis, L. (2009). Fitness functions in evolutionary robotics: A survey and analysis. *Robot. Auton. Syst.*, 57(4), 345–370.
- Pinciroli, C., Trianni, V., O’Grady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Di Caro, G., Ducatelle, F., Birattari, M., Gambardella, L., and Dorigo, M. (2012). ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems. *Swarm Intelligence*, 6(4), 271–295.
- Rabin, M.O. (1963). Probabilistic Automata. *Information and Control*, 6(3), 230–245.
- Şahin, E. (2004). Swarm robotics: From sources of inspiration to domains of application. In E. Şahin and W.M. Spears (eds.), *Swarm Robotics, SAB*, volume 3342 of *LNCS*, 10–20. Springer, Berlin Heidelberg, Germany.
- Urzalai, J. and Floreano, D. (2000). Evolutionary robotics: coping with environmental change. In L.D. Whitney and et al. (eds.), *Proceedings of Conference on the Genetic and Evolutionary Computation Conference, GECCO*, 941–948. Morgan Kaufmann, San Francisco CA.
- Watson, R., Ficici, S., and Pollack, J. (2002). Embodied evolution: distributing an evolutionary algorithm in a population of robots. *Robotics and Autonomous Systems*, 39(1), 1–18.
- Zagal, J.C. and Ruiz-Del-Solar, J. (2007). Combining simulation and reality in evolutionary robotics. *Journal of Intelligent & Robotic Systems*, 50(1), 19–39.