Monitoramento de Recursos para Aplicações de Robótica em Espaços Inteligentes baseados em uma Nuvem OpenStack

Pablo B. Santos[1], João H. G. Medeiros[1], Rodolfo Picoreti[2], Felippe M. de Queiroz[2] Diego G. Cardoso[1], Alexandre P. do Carmo[2],Raquel F. Vassalo[2], Moises R. N. Ribeiro[2], Rodolfo S. Villaça[1]

[1]Programa de Pós-Graduação em Informática (PPGI)
Universidade Federal do Espírito Santo (UFES)

[2]Programa de Pós-Graduação em Engenharia Elétrica (PPGEE)
Universidade Federal do Espírito Santo (UFES)

pablobrunetti@hotmail.com,{jhenrique,rodolfo.picoreti}@gmail.com
{mendonca.felippe,dgiacomellic}@gmail.com, alexandre.carmo@ifes.edu.br
{raquel,moises}@ele.ufes.br, rodolfo.villaca@ufes.br

Abstract. *Cloud Computing environments require constant measurement and monitoring to ensure the proper functioning of the different applications hosted in the cloud. Questions such as "How to do this?", "What?" and "How to integrate application and cloud performance metrics?" are relevant aspects that need to be investigated. This paper presents a case study that describes the instrumentation of an OpenStack cloud environment that hosts a robot-trajectory control application in an intelligent space. Different resource allocation configurations were evaluated, which correspond to different orchestration decisions, and the result is evaluated as a function of the robot-trajectory error. The results show that good orchestration decisions yields better results without overloading the cloud with the allocation of unnecessary resources to the application.*

Resumo. *As nuvens computacionais necessitam de medição e monitoramento constante para garantir o funcionamento adequado das diferentes aplicações ali hospedadas. Questões como: "Como monitorar?", "O quê monitorar?" e "Como integrar métricas de desempenho das aplicações à nuvem?" são aspectos relevantes que precisam ser investigados. Este artigo apresenta um estudo de caso que descreve a instrumentação de uma nuvem computacional OpenStack que hospeda uma aplicação de controle de trajetória de um robô em um espaço inteligente. Foram avaliadas diferentes configurações de alocação de recursos, que correspondem a diferentes decisões de orquestração e o resultado é avaliado em função do erro de trajetória do robô. Os resultados mostram que a correta alocação de recursos gera melhores resultados sem sobrecarregar a nuvem com a alocação de recursos desnecessários.*

1. Introdução

Ambientes de Computação em Nuvem têm sido amplamente disponibilizados por grandes, pequenos e médios provedores de serviço. Um dos ambientes mais populares é o IaaS (*Infrastructure as a Service*), onde recursos de computação são disponibilizados para que os usuários hospedem suas aplicações nessas infraestruturas compartilhadas. No

âmbito acadêmico, vários ambientes experimentais têm sido implantados usando conceitos e recursos de nuvens computacionais. Dentre esses ambientes destaca-se o FUTEBOL (*Federated Union of Telecommunications Research Facilities for an EU-Brazil Open Laboratory*) [1], projeto no qual a proposta deste artigo está vinculada.

Da mesma forma que as infraestruturas de redes e de servidores tradicionais requerem constante medição e monitoramento de redes e serviços para garantir o correto dimensionamento dos recursos disponíveis, as nuvens computacionais também têm essa necessidade, com o agravante de que a diversidade de diferentes níveis de serviço que precisam ser atendidos é bem maior, o que torna a tarefa bastante desafiadora [Faniyi and Bahsoon 2015].

Subsidiar os orquestradores de recurso nas nuvens computacionais com informações adequadas, de tal forma a garantir a correta alocação dos recursos para os clientes e aplicações e garantir um nível de qualidade de experiência (QoE) para os clientes da nuvem computacional continua sendo um tema de grande relevância, tanto no mercado, quanto na academia. "Como monitorar?", "O quê monitorar?" e "Como integrar métricas de desempenho das aplicações à nuvem?" são perguntas relevantes nesse contexto e que precisam ser investigadas para serem respondidas adequadamente.

Sendo assim, a proposta deste artigo é apresentar um estudo de caso, realizado no contexto do projeto FUTEBOL, que descreve a instrumentação de uma nuvem computacional OpenStack que hospeda uma aplicação de controle de trajetória de um robô em um espaço inteligente [Queiroz et al. 2015, dos Santos et al. 2017] para fins de medição e monitoramento. Nessa aplicação o processamento das funções de controle do robô foi migrado para um ambiente de nuvem computacional no *datacenter* do Nerds (Núcleo de Estudos em Redes Definidas por Software), enquanto o robô e o espaço inteligente permaneceram em outro prédio, no VIROS (*Vision and Robotics Systems*), ambos na Universidade Federal do Espírito Santo (Ufes).

Nesse contexto, este artigo tem dois objetivos principais. O primeiro é mostrar, por meio de uma nuvem OpenStack, recursos de medição e monitoramento e como a integração de diferentes ferramentas viabiliza a aquisição de diversas métricas de desempenho, tanto da nuvem quanto da aplicação. Além disso, o segundo objetivo é mostrar que a utilização adequada destas métricas facilita o trabalho de orquestração e produz resultados significativos na Qualidade de Experiência dos usuários em relação às aplicações hospedadas na nuvem.

Por meio de experimentos realizados em laboratório foram avaliadas diferentes configurações de alocação de recursos para os serviços que compõem a aplicação de controle do robô no espaço inteligente. Para cada configuração foram medidas e monitoradas as seguintes métricas: utilização de CPU e memória, além da latência de execução de cada serviço. Neste artigo aspectos referentes ao papel da rede no desempenho da aplicação não serão considerados. As diferentes configurações correspondem a diferentes decisões de alocação de recursos e o resultado final é sempre avaliado em função do erro de trajetória do robô. Os resultados mostram que a correta alocação de recursos gera os melhores resultados sem sobrecarregar a nuvem com a alocação de recursos desnecessários (ociosos). Em [Gomes et al. 2017] foi mostrada a primeira fase do projeto onde

[1] http://www.ict-futebol.org.br/

a aplicação foi executada em um contexto local, em servidores dentro do próprio laboratório VIROS. É possível observar no artigo alguns gargalos em sua execução, porém por não estar em uma infraestrutura em nuvem e não possuir o adequado monitoramento do ambiente, ainda não era possível identificar e alocar os recursos adequadamente.

O restante deste artigo encontra-se organizado da seguinte forma: a Seção 2 faz um apanhado de diversos trabalhos relacionados às áreas de orquestração e monitoramento de recursos na nuvem, espaços inteligentes e robótica na nuvem. Em seguida, na Seção 3 a aplicação de controle de um robô em um espaço inteligente é descrita, com todos os seus módulos e interações. A Seção 4 descreve o ambiente e as ferramentas de monitoramento usadas na nuvem para o desenvolvimento deste artigo. Os testes e avaliação dos resultados obtidos são descritos na Seção 5 e a Seção 6 conclui o artigo e aponta possíveis trabalhos futuros e novas contribuições.

2. Trabalhos Relacionados

Robótica na nuvem é uma área em bastante crescimento. Atualmente há diversos trabalhos que estudam a integração entre robótica e computação na nuvem, tais como [Miratabzadeh et al. 2016, Benavidez et al. 2015]. [Miratabzadeh et al. 2016], por exemplo, propõe uma arquitetura de software para robótica na nuvem implementada na Infraestrutura como Serviço em uma nuvem OpenStack. Entretanto, a maioria dos trabalhos na área focam na descrição dos sistemas de robótica, enquanto que este artigo foca nos aspectos de medição e monitoramento de aplicações de robótica na nuvem.

Em [Benavidez et al. 2015] é apresentado um sistema e algoritmos de *Visual Slam (VSLAM)* que reduzem o tempo de processamento em tempo real de robôs e os requisitos para se realizar todo o processamento na nuvem OpenStack. Entretanto os autores não abordam nenhum aspecto relacionado ao monitoramento da nuvem OpenStack ao executar as aplicações propostas por eles.

Em [Almonfrey et al. 2018] é apresentado um detector de pedestres flexível desenvolvido para ser executado em um espaço inteligente com uma rede multicâmeras. Dentre os estudos de caso apresentados, há um controle de um robô para desvio de pessoas dentro do ambiente. Este artigo mostra que o serviço de detecção de pedestre foi desenvolvido de forma a tirar proveito das capacidades que um ambiente em nuvem pode prover, tal como escalar verticalmente e horizontalmente. Porém não é abordada a necessidade do monitoramento e a definição de métricas para esta escalada.

Em [Bezerra et al. 2014] é apresentado o MAS (*Automatic Migration and Allocation System for Virtual Resource Management*), que monitora recursos de máquinas físicas e virtuais e constrói perfis de uso detectando, com base neste perfis, a escassez dos recursos monitorados. O MAS automaticamente redistribui as cargas de trabalho evitando a sobrecarga dos nós. Entretanto, o MAS utililiza a biblioteca `libvirt`, plataforma de Virtualização Xen e seus resultados não correspondem ao monitoramento de uma aplicação real, como neste artigo.

Em [Bruschi et al. 2016] é apresentado o avaliador de desempenho de uma nuvem IaaS multicamada chamada StackAckt. O StackAckt é um mecanismo que permite monitorar e obter dados sobre o consumo de recursos computacionais em três camadas utilizando o orquestrador *IaaS* Apache CloudStack, o *hipervisor* XenServer e armazena-

mento de dados no sistema OpenFiler. Embora descreva soluções de medição e monitoramento de nuvens, o trabalho o faz usando CloudStack e XenServer, sem utilizar a nuvem OpenStack para realizar o monitoramento dos recursos computacionais.

Baseando-se nos trabalhos relacionados apresentados nesta seção, até onde tem-se conhecimento, no contexto de robótica na nuvem, espaços inteligentes, medição e monitoramento de nuvens computacionais, este é o primeiro trabalho que integra esses assuntos em uma abordagem prática-experimental.

3. Robótica na Nuvem e Espaços Inteligentes

Atualmente há uma nova geração de aplicações de robótica, denominadas "robótica como serviço" em espaços inteligentes, que utilizam controle em tempo real, remoto, e são aplicadas em diversas áreas, tais como terapia de reabilitação, localização e navegação de robôs e robótica social. Espaços inteligentes podem ser descritos como locais que tem sensores conectados a rede (câmeras e microfones, por exemplo) que coletam informações do ambiente e repassam a um sistema supervisor que analisa as informações, habilitam tomadas de decisão, interações entre usuários e a execução de uma ação de um dispositivo conectado a rede (robôs, dispositivos móveis, monitor de informações) [Hvizdoš et al. 2017].

A Figura 1 apresenta a arquitetura física do espaço inteligente, na qual pode-se notar que no ambiente que o robô se encontra há apenas o necessário para este realizar uma tarefa de seguimento de trajetória: câmeras que realizam a aquisição das imagens utilizadas para a localização e uma rede sem fio utilizada para enviar comandos de velocidade e direção para o robô. Na figura os componentes que realizam o processamento das imagens e determinam os comandos de trajetória do robô estão todos alocados na nuvem, ou seja, em *datacenters* remotos.

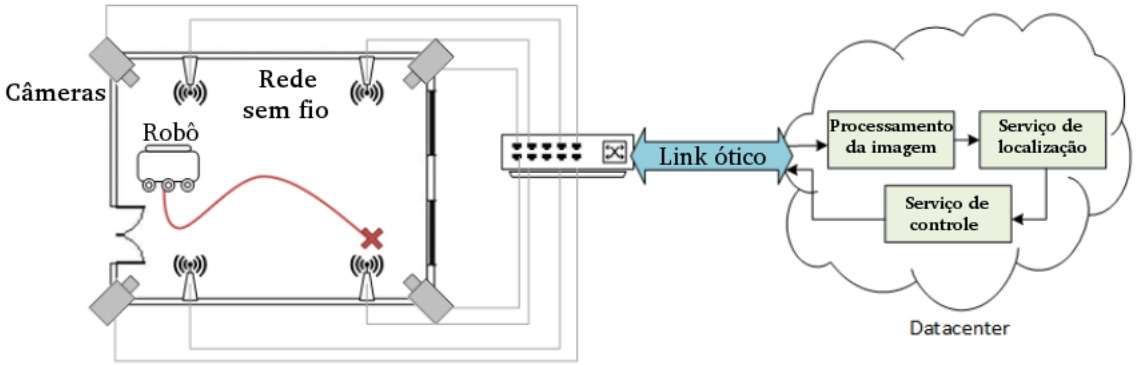


Figura 1. Descrição do cenário de experimento.

A tarefa de seguimento de trajetória consiste em manter o robô em uma dada posição no espaço-tempo com uma determinada velocidade. Por conta disso, dado uma taxa de amostragem fixa das câmeras, há o requisito de se computar as etapas de processamento de imagem para a localização, e cálculo da velocidade, além do envio dos comandos de controle para o robô, em um tempo menor ou igual ao período de amostragem.

Portanto, é importante que a nuvem dê recursos suficientes para os componentes da aplicação de controle do robô de forma que o requisito de tempo de resposta seja atendido. Para isso a orquestração deve levar em consideração a capacidade de processamento oferecida a cada serviço que compõe a aplicação, com o objetivo de diminuir o seu tempo de processamento, bem como o tempo de comunicação entre esses serviços. A soma de todos esses tempos resulta no tempo para envio do comando do controle do robô. Caso o comando de controle atrase, o robô continuará executando o comando anterior fazendo com que ele saia da trajetória planejada, e, consequentemente aumente o erro de seguimento.

Além disso, o erro de trajetória nesta tarefa está associado à taxa de amostragem utilizada. Quanto maior esta for, menor será o erro de trajetória a depender também da velocidade que o robô a executa. Portanto, além do requisito de latência no tempo de resposta dos serviços, para que haja um baixo erro de trajetória aumentando a taxa de amostragem, há o requisito de largura de banda referente à transmissão das imagens de alta resolução à nuvem. Desta maneira, quanto maior a taxa de quadros utilizada, maior será a quantidade de recursos requisitados à nuvem para que se consiga executar a tarefa com o menor erro de trajetória possível.

O funcionamento básico da aplicação é apresentado na Figura 2, relacionando o espaço inteligente e os serviços remotos com o processamento da imagem, o controle e localização do robô. É mostrado também a utilização das formas de transmissão de dados via cabo (fibra ótica) e sem fio no processamento da aplicação.

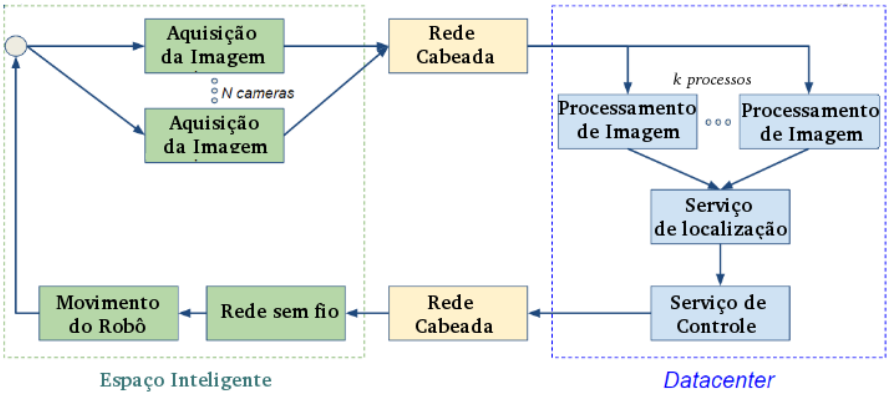


Figura 2. Diagrama de funcionamento do experimento.

4. Monitoramento do Ambiente OpenStack

O monitoramento do *setup* do espaço inteligente no *datacenter* foi realizado através da plataforma Infraestrutura como serviço (*IaaS*) OpenStack *multi-node* na versão Pike [2]. O OpenStack é um sistema operacional de nuvem que controla grandes conjuntos de recursos de computação, armazenamento, e recursos de rede dentro de um *datacenter*. Ele é dividido em diversos módulos que possuem diferentes funções no ambiente de nuvem. Neste trabalho, foram utilizados os seguintes módulos: Nova (provisionamento de instâncias de computação), Keystone (gerência de acesso à nuvem), Glance (gerência de

[2]https://www.openstack.org/software/pike/

imagens), Neutron (conectividade entre as redes virtuais), Horizon (painel de controle da nuvem via página *web*) e Ceilometer (coleta de dados para monitoramento).

Para realizar o monitoramento foi utilizado o módulo Ceilometer, que é um serviço de coleta de dados que oferece a capacidade de medir métricas de desempenho e eventos de todos os componentes do OpenStack. Para o armazenamento e posterior recuperação de dados foi utilizado um serviço de banco de dados temporal integrado à nuvem, o Gnocchi [3]. O Gnocchi é um banco de dados de código aberto, que permite o armazenamento e indexação de dados e recursos de séries temporais em grande escala. Isso é útil em plataformas de nuvem modernas que não são apenas de grande dimensão, mas também são dinâmicas e potencialmente multiusuário.

Para a instalação do ambiente em Nuvem OpenStack foi utilizada a infraestutura com 1 (um) nó responsável por controlar todas as ações realizadas no OpenStack e 2 (dois) nós responsáveis por prover as instâncias das máquinas virtuais que hospedariam os diferentes serviços da aplicação de controle, na nuvem, da trajetória de um robô em um espaço inteligente. A configuração das três máquinas por prover a nuvem OpenStack são computadores Dell Power EDGE T430 com 16GB de memória RAM e CPU Intel Xeon E5-2603 com 12 cores.

A infraestrutura de máquinas virtuais foi criadas de acordo com a Figura 3 e é composta por 1 (um) serviço sincronizador de tempo entre as câmeras (Time.Sync), 4 (quatro) *gateways* das Cameras (Camera Gateway) que se comunicam com as câmeras e as configuram de acordo com a necessidade da aplicação, 1 (um) serviço de troca de mensagens (RabbitMQ) [4], 1 (um) controlador do robô (Robot Controller) e n serviços de processamento de imagens e localização do robô (ArUco) que podem ser paralelizados. Todos os módulos descritos aqui referem-se à Figura 2.

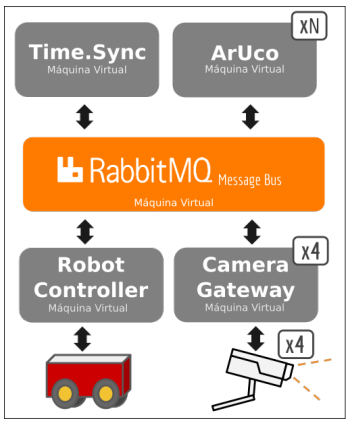


Figura 3. Infraestrutura de máquinas virtuais criadas no OpenStack.

5. Testes e Avaliação dos Resultados

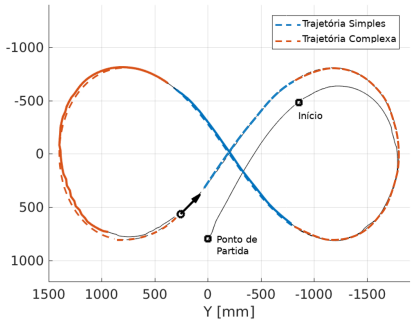
Na configuração dos experimentos foram definidos 3 (três) diferentes cenários, sendo que em cada um há uma variação de configuração dos recursos computacionais das máquinas virtuais alocadas ao serviço de processamento de imagens e localização do robô (ArUco).

[3]https://gnocchi.xyz/
[4]https://www.rabbitmq.com/

Todos os serviços que rodam nas máquinas virtuais e os computadores físicos foram monitorados ao longo do experimento. O ArUco é o serviço que possui o maior tempo de resposta entre aqueles que compõe a aplicação de controle do robô, e por isso os resultados focarão nas métricas e configurações de suas máquinas virtuais.

Na Figura 4(a) é mostrado o robô no espaço inteligente do Laboratório VIROS a partir da imagem de uma das câmeras. Em todos os cenários foi realizada uma trajetória na forma de Lemniscata de Bernoulli conforme Figura 4(b). A trajetória se divide em duas partes denominadas trajetória simples (Linha reta com velocidade constante) e trajetória complexa (Caminho curvo com acelerações positivas e negativas). O robô inicia a trajetória no ponto marcado como ponto de partida, porém seus resultados são validados a partir do ponto início, uma vez que a posição inicial do robô não coincide com a da trajetória, o que faz com que o erro inicial seja alto mesmo estando em um trecho de trajetória fácil de seguir. Sendo assim, esta parte inicial é eliminada dos resultados para que não aumente o valor médio do erro analisado durante a execução da tarefa. O robô realiza então 3 voltas com duração de 25 segundos cada.

(a) Robô no Laboratório VIROS



(b) Trajetória na forma de Lemniscata de Bernoulli

Figura 4. Execução do robô no espaço inteligente.

5.1. Cenário 1

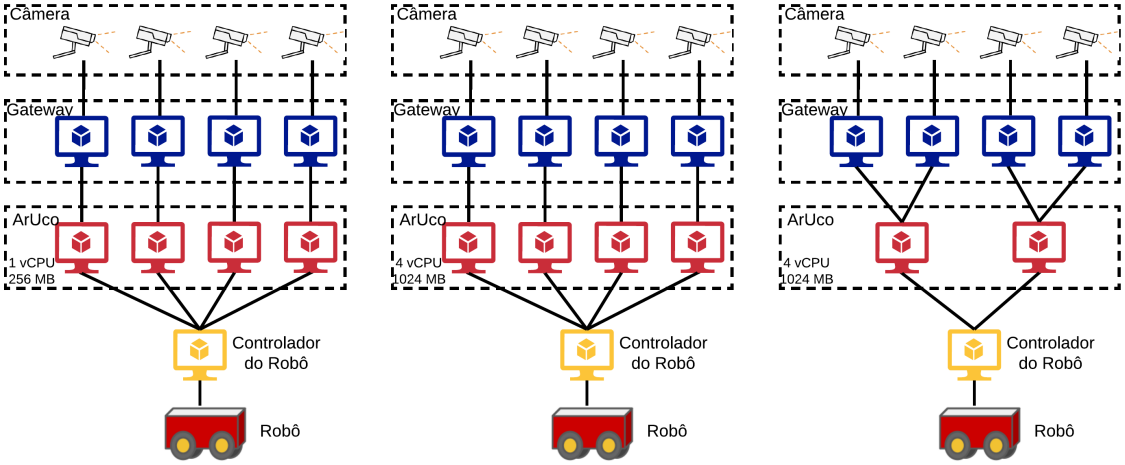
Neste cenário tem-se a infraestrutura de máquinas Virtuais como mostrada na Figura 5(a), com 4 (quatro) máquinas virtuais ArUco configuradas com 1 (uma) vCPU e 256MB de memória RAM. Nesse cenário foram atribuídos recursos computacionais bem limitados para prover o processamento do robô.

5.2. Cenário 2

Neste cenário tem-se a infraestrutura de máquinas Virtuais como mostrada na Figura 5(b), com a quantidade de máquinas virtuais ArUco permanecendo como 4 e aumento dos recursos computacionais para 4 vCPUs e 1GB de memória RAM. Nesse cenário foi atribuído um alto poder computacional para todas as máquinas virtuais que realizam o processamento do robô.

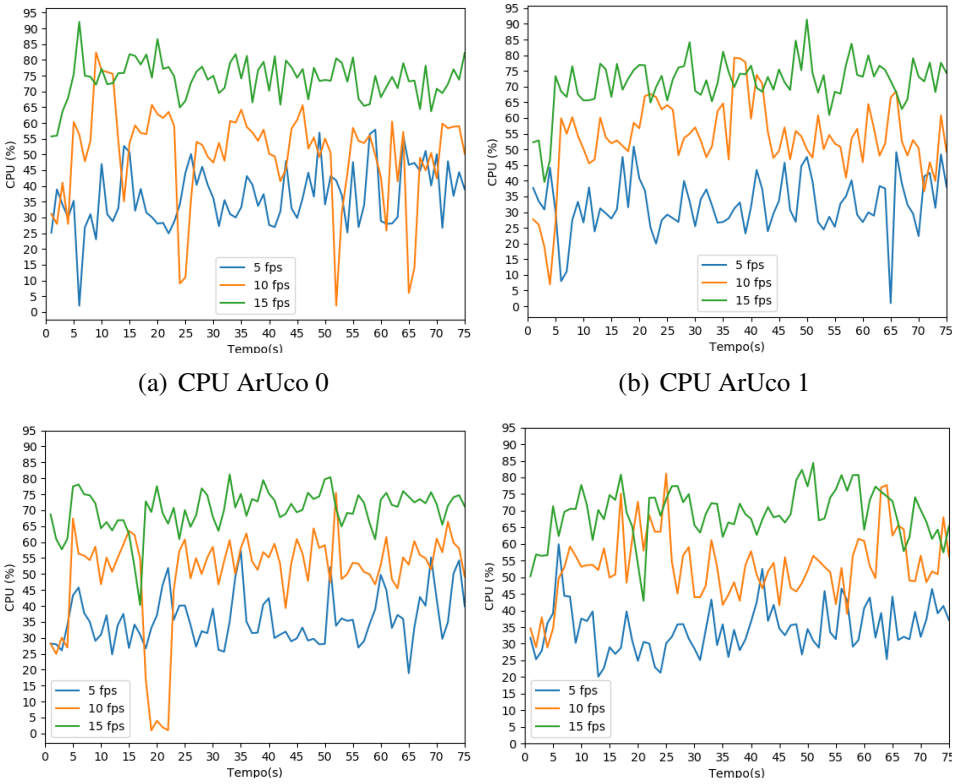
5.3. Cenário 3

Neste cenário temos a infraestrutura de máquinas Virtuais como mostrada na Figura 5(c), reduzindo a quantidade de máquinas virtuais ArUco para 2, ou seja, o processamento de imagens das 4 câmeras são realizadas por 2 máquinas virtuais ArUco. As configurações das 2 ArUco permanecem como no cenário anterior (4 vCPUs e 1GB de memória RAM).



(a) Cenário 1 (b) Cenário 2 (c) Cenário 3

Figura 5. Diagrama dos diferentes cenários utilizados nos experimentos.



(a) CPU ArUco 0 (b) CPU ArUco 1

(c) CPU ArUco 2 (d) CPU ArUco 3

Figura 6. Taxa de processamento das máquinas ArUco no Cenário 1.

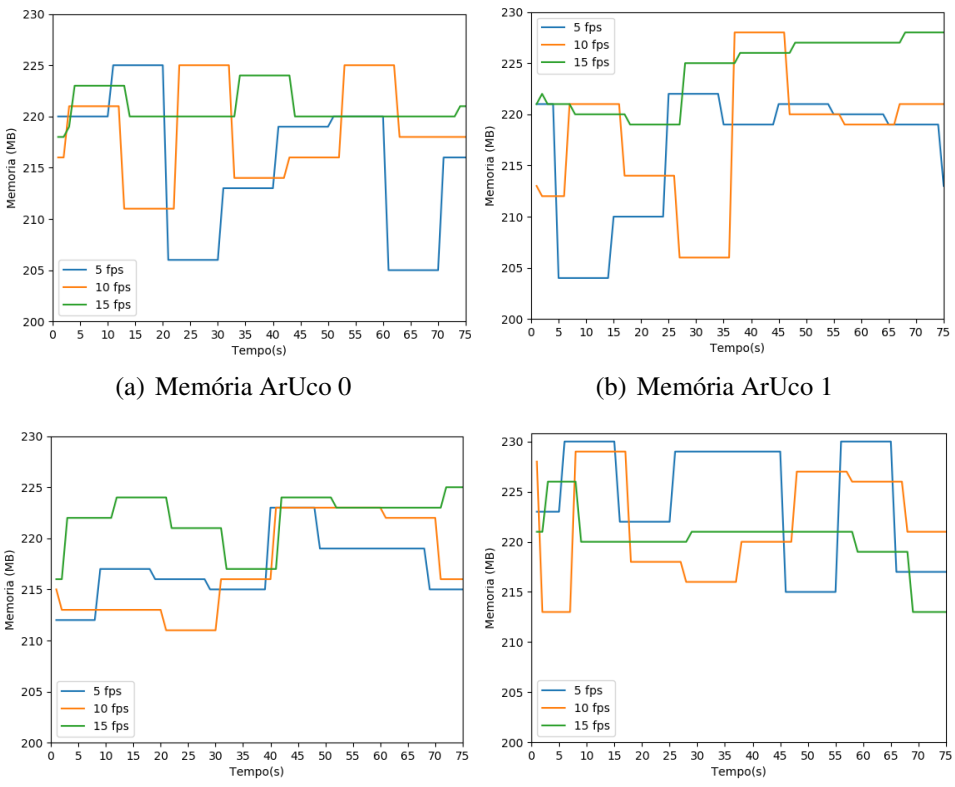
5.4. Orquestração dos Recursos

Todos os testes foram realizados com configurações de 5, 10 e 15 *frames* por segundo (fps) nos 3 (três) cenários. Para cada fps tem-se uma janela de tempo de processamento distinta, que correspondem à, respectivamente, 200, 100 e 66.67 ms. Isso ocorre pois, no

decurso destas janelas, novas imagens serão enviadas pelas câmeras para processamento na nuvem e a imagem atual (em processamento) precisará ser descartada. Os descartes impactam significativamente no erro de trajetória do robô, conforme descrito na Seção 3. Ou seja, intuitivamente, quanto maior o fps a ser processado na nuvem, maior será a quantidade de recursos computacionais utilizados, tais como memória RAM, utilização da CPU e largura de banda usada na rede. Em todos os cenários apresentados serão analisados as métricas de utilização de memória e CPU. O monitoramento de uso da rede foi efetuado, porém será apresentado em trabalhos futuros pois este não se mostrou um gargalo no experimento, considerando as taxas de fps utilizadas.

No Cenário 1 temos, nas Figuras 6(a), 6(b), 6(c) e 6(d), o resultado do monitoramento da taxa de utilização da CPU durante o experimento com fps igual a 5, 10 e 15. Nesse cenário é possível visualizar que a taxa média da utilização de CPU fica em torno de 75% na Figura 6(a) e 70% na Figura 6(b) e, quando efetuado o teste com a taxa de 15 fps, chega a alcançar picos de 90% (Figuras 6(a) e 6(b)).

Com relação a utilização de memória RAM podemos visualizar, nas Figuras 7(a), 7(b), 7(c) e 7(d), o resultado do monitoramento durante o experimento. É fácil notar um grande consumo de memória em todas as ArUcos, chegando a gastar até 230MB em 7(d), sendo que as VMs, neste cenário, possuem um limite de 256MB de memória RAM (utilização de 89.8%). Pode-se concluir que é necessário realizar uma nova alocação de recursos computacionais para reduzir esses gargalos e obter resultados melhores no erro de trajetória do robô.

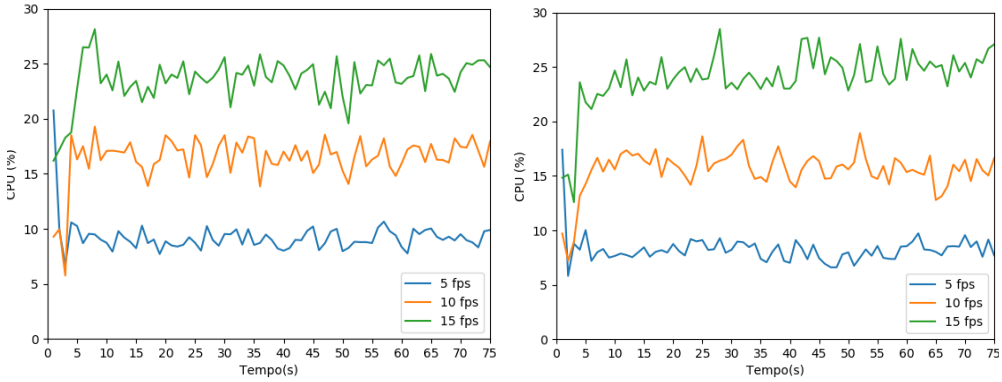


(a) Memória ArUco 0 (b) Memória ArUco 1

(c) Memória ArUco 2 (d) Memória ArUco 3

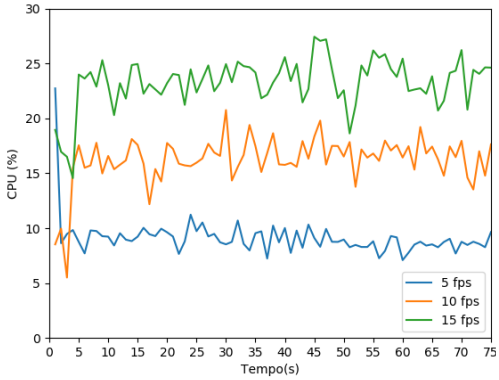
Figura 7. Quantidade de memória utilizada das máquinas ArUco no Cenário 1.

As Figuras 8(a), 8(b), 8(c) e 8(d) apresentam os resultados do monitoramento da utilização de CPU das ArUco com 5, 10 e 15 fps no Cenário 2. Relembrando, neste cenário foram acrescidos mais recursos computacionais às máquinas ArUco, incrementando a memória RAM de 256MB para 1GB e a quantidade de vCPUs de 1 para 4. Na análise temos uma diminuição da média de utilização de CPU em quase 3 (três) vezes em todas as ArUco, quando comparadas ao Cenário 1. Com relação a quantidade de memória utilizada é possível visualizar, nas Figuras 9(a), 9(b), 9(c) e 9(d) que escalar verticalmente o recurso de memória de 256MB para 1024MB fez com que a utilização de memória não configurasse um gargalo para as máquinas ArUco.

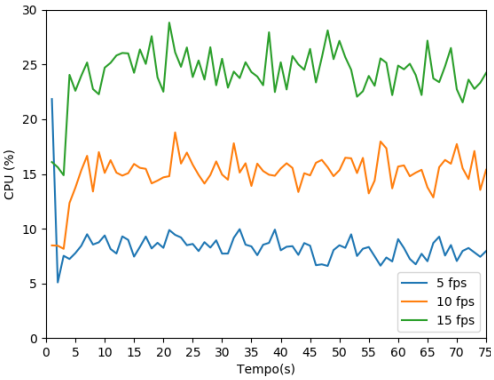


(a) CPU ArUco 0

(b) CPU ArUco 1



(c) CPU ArUco 2

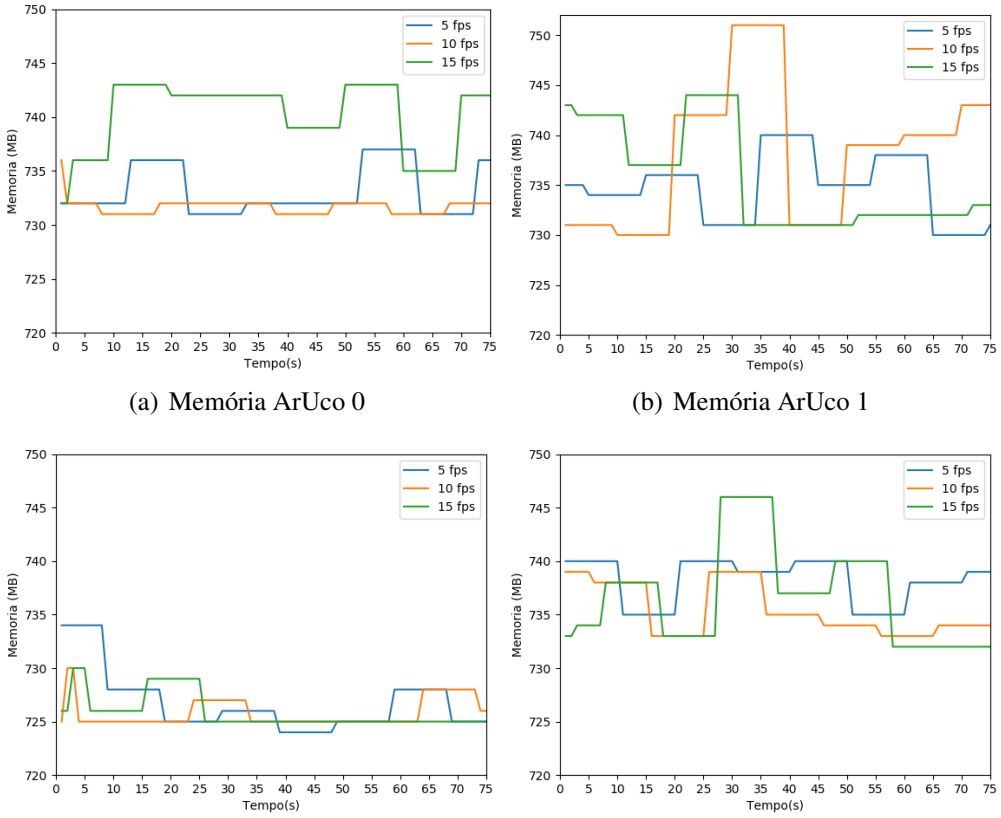


(d) CPU ArUco 3

Figura 8. Taxa de processamento das máquinas ArUco no Cenário 2.

No Cenário 3 a quantidade de máquinas ArUco foi reduzida para 2. É possível perceber, nas Figuras 10(a), 10(b), que ocorreu um leve aumento da utilização da CPU em comparação ao Cenário 2. A quantidade de memória utilizada no experimento é ilustrada nas Figuras 11(a) e 11(b) e não teve alterações significativas em relação ao Cenário 2.

Além de monitorar as máquinas virtuais e os recursos da nuvem é realizado também o monitoramento da aplicação integrado ao OpenStack como mostrado na Figura 12. Um exemplo do tempo de processamento dos serviços que compõe uma aplicação pode ser observado na Figura 12, sendo possível observar que o processamento está ocorrendo dentro da janela de 67 ms correspondente a 15 fps. O monitoramento dos tempos de processamento de cada serviço é realizado pela própria aplicação e enviado à infraestrutura.



(a) Memória ArUco 0 (b) Memória ArUco 1

(c) Memória ArUco 2 (d) Memória ArUco 3

Figura 9. Quantidade de memória utilizada das máquinas ArUco no Cenário 2.

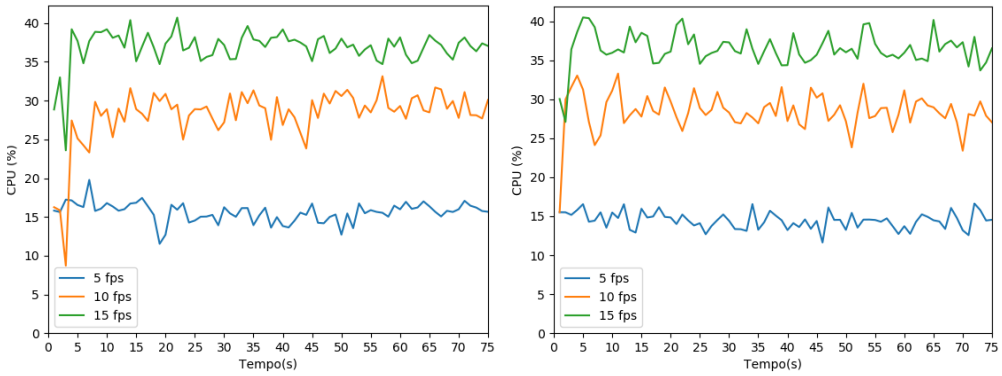
5.5. Avaliação do Erro de Trajetória

Na Figura 13 é apresentado o erro de trajetória em forma de *boxplot* em todos os cenários. A taxa escolhida foi de 15 fps, por ser a taxa que demanda mais recursos computacionais da nuvem dentre os 3 (três) cenários. No Cenário 1 temos um alto erro de trajetória, no conjunto de trajetória fácil e difícil, em comparação com os Cenários 2 e 3. No Cenário 2, o erro da trajetória é baixo porém com um alto consumo de recursos. Já no Cenário 3, há uma redução significativa do uso de recursos ao utilizar metade das máquinas virtuais, mas ainda assim mantem-se o erro de trajetória em níveis similares ao Cenário 2. Desta forma, o Cenário 3 mostra-se como a escolha mais adequada tanto do ponto de vista da aplicação, através da medida do erro da trajetória, quanto do ponto de vista da infraestrutura, através da utilização dos recursos.

A partir desses resultados é possível afirmar que existe uma clara necessidade de se orquestrar os recursos computacionais disponíveis a fim de se obter os melhores resultados possíveis para a aplicação hospedada na nuvem.

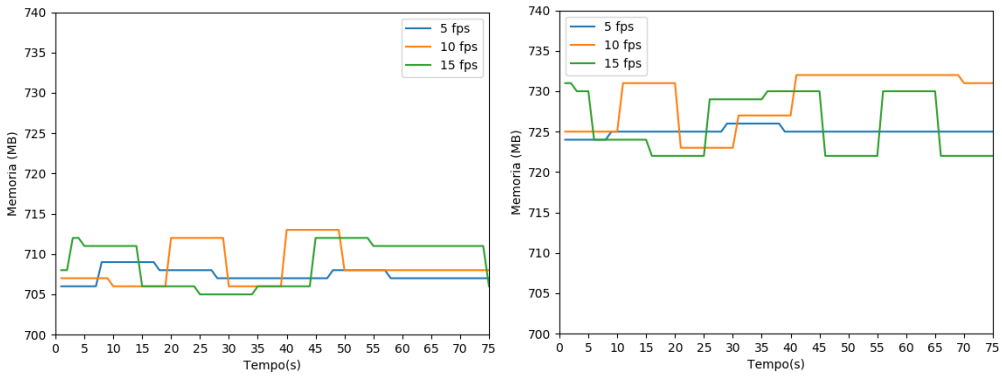
6. Conclusões e Trabalhos Futuros

O presente trabalho apresentou, por meio de uma nuvem OpenStack, recursos de medição e monitoramento, integrado com métricas de desempenho, visando uma melhor orquestração dos recursos da nuvem e uma maior utilização de aplicações hospedadas na nuvem.



(a) CPU ArUco 0 (b) CPU ArUco 1

Figura 10. Taxa de processamento das máquinas ArUco no Cenário 3.



(a) Memória ArUco 0 (b) Memória ArUco 1

Figura 11. Quantidade de memória utilizada das máquinas ArUco no cenário 3.

Para tanto, foram realizados experimentos em laboratório, avaliando, em cenários distintos, as consequências de não utilizar recursos computacionais adequados às aplicações. Ou seja, o provisionamento de recursos deve se adequar a necessidade do que está sendo executado na nuvem, sem criar um gargalo computacional devido ao provisionamento de poucos recursos, ou um desperdício de recursos (e consequentemente sobrecarregando a nuvem) com poder computacional muito acima da necessidade da aplicação.

Foi verificado que, para a aplicação do robô no espaço inteligente, quando os recursos foram escalados (saindo de um cenário com uma vCPU e 256 MB de memória para o cenário de quatro vCPUs e 1 GB de memória) houve um ganho para a execução da aplicação, diminuindo os erros de trajetória do robô. Por outro lado, quando houve a diminuição da quantidade de máquinas virtuais responsáveis pelo processamento da imagem e localização do robô no espaço (saindo de quatro VMs para duas, com quatro vCPUs e 1 GB de memória), visando uma diminuição da sobrecarga da nuvem, verificou-se que não há diferença significativa no erro de trajetória do robô. Ou seja, para o relacionamento entre a nuvem e aplicação, o último cenário faz mais sentido, já que há uma economia de recursos computacionais, enquanto que não há diferença sensível para a aplicação.

Portanto, é necessário que haja uma orquestração dos recursos, oferecendo um poder computacional adequado a atender as especifidades da aplicação executada na nu-

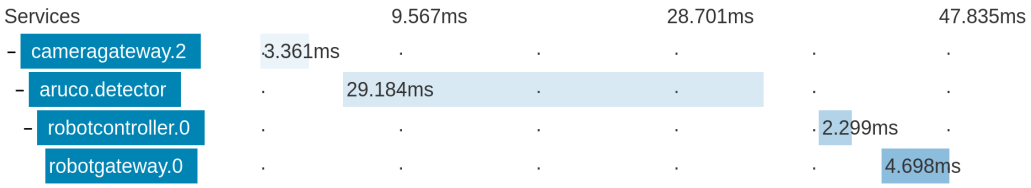


Figura 12. Visualização do *tracing* de um ciclo de controle do robô utilizando a ferramenta Zipkin, para uma taxa de amostragem de 15 fps.

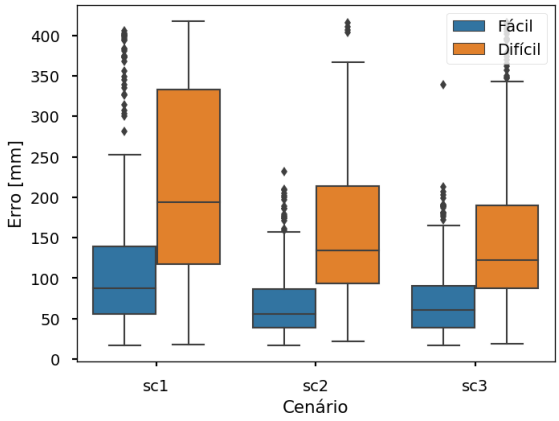


Figura 13. Erro de trajetória nos 3 Cenários com 15 fps.

vem. Dessa forma, como trabalhos futuros pretende-se desenvolver um orquestrador de recursos da nuvem OpenStack, provisionando recursos de forma automática, de acordo com a demanda da aplicação.

Outro trabalho futuro está na verificação do grau de comprometimento da rede. Ou seja, pretende-se aumentar a demanda, por meio de aumento de fps por exemplo, ou alterar a largura de banda. Com essas alterações pode-se verificar o quanto o estado da rede influencia em uma aplicação executada na nuvem. Se o fato de haver um congestionamento ou uma maior largura de banda alterar a forma com que a aplicação realiza sua execução, torna-se necessário orquestrar recurso de rede por meio da utilização de redes programáveis do tipo SDN (*Software Defined Networking*), por exemplo.

Agradecimentos

Este trabalho recebeu financiamento parcial do projeto Horizon 2020 (União Européia) sob no. 688941 (FUTEBOL), assim como do MCTI por meio da RNP e do CTIC. Além disso, os autores gostariam de agradecer o financiamento parcial proveniente de recursos de bolsas e projetos CNPq, CAPES e FAPES.

Referências

Almonfrey, D., do Carmo, A. P., de Queiroz, F. M., Picoreti, R., Vassallo, R. F., and Salles, E. O. T. (2018). A flexible human detection service suitable for intelligent spaces based on a multi-camera network. *International Journal of Distributed Sensor Networks*, 14(3):1550147718763550.

Benavidez, P., Muppidi, M., Rad, P., Prevost, J. J., Jamshidi, M., and Brown, L. (2015). Cloud-based realtime robotic visual slam. In *2015 Annual IEEE Systems Conference (SysCon) Proceedings*, pages 773–777.

Bezerra, G. M. G., Mattos, D. M. F., Ferraz, L. H. G., and Duarte, O. C. M. B. (2014). Sistema automatizado de gerência de recursos para ambientes virtualizados. In *Anais do 32o Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos – SBRC 2014*, pages 677–690.

Bruschi, G. C., Spolon, R., Pauro, L. L., Lobato, R. S., Manacero, A., and Cavenaghi, M. A. (2016). Stackact: Performance evaluation in an iaas cloud multilayer. In *2016 15th International Symposium on Parallel and Distributed Computing (ISPDC)*, pages 149–156.

dos Santos, C. C., Picoreti, R., do Carmo, A. P., Vassallo, R. F., and Garcia, A. S. (2017). Modelagem de um comportamento interacional entre homem e robô para um espaço inteligente baseado em visão computacional. In *Simpósio Brasileiro de Automação Inteligente*, SBAI 2017, Porto Alegre/RS. SBA.

Faniyi, F. and Bahsoon, R. (2015). A systematic review of service level management in the cloud. *ACM Comput. Surv.*, 48(3):43:1–43:27.

Gomes, R. L., Martinello, M., Dominicini, C. K., Hasse, P., Villaca, R., Vassallo, R. F., do Carmo, A. P., de Queiroz, F. M., Picoreti, R., Garcia, A. S., Ribeiro, M. R. N., Espin, J. A. G., Hammad, A., Nejabati, R., and Simeonidou, D. (2017). How can emerging applications benefit from eaas in open programmable infrastructures? In *IEEE SUMMER SCHOOL ON SMART CITIES*, IEEE SSSC 2017, Natal/RN. IEEE.

Hvizdoš, J., Vojtko, I., Koscelanský, M., Pavlov, J., Vaščák, J., and Sinčák, P. (2017). Applications of remote controlled robotics in the intelligent space. In *2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pages 117–122.

Miratabzadeh, S. A., Gallardo, N., Gamez, N., Haradi, K., Puthussery, A. R., Rad, P., and Jamshidi, M. (2016). Cloud robotics: A software architecture: For heterogeneous large-scale autonomous robots. In *2016 World Automation Congress (WAC)*, pages 1–6.

Queiroz, F. M., Covre, V. B., Rampinelli, M., Pereira, F. G., Vassallo, R. F., and Filho, T. F. B. (2015). Localização e guiagem de um robô móvel em um espaço inteligente. In *Simpósio Brasileiro de Automação Inteligente*, SBAI 2015, Natal/RN. SBA.