Performance Evaluation of 3D Reconstruction Tool Over Cloud Computing Virtualized Environment

Article *in* IEEE Latin America Transactions · October 2016

DOI: 10.1109/TLA.2016.7786319

3 authors, including:

David Beserra
Université de Paris 1 Panthéon-Sorbonne
27 PUBLICATIONS **17** CITATIONS

SEE PROFILE

Patricia Takako Endo
University of Pernambuco
27 PUBLICATIONS **217** CITATIONS

SEE PROFILE

Performance Evaluation of 3D Reconstruction Tool Over Cloud Computing Virtualized Environment

G. Leoni, D. W. Beserra and P. T. Endo

***Abstract*— The Creative Cities movement looks for creative solutions to environmental issues, or inventive ways for people. One of the main objectives of the Creative Cities is to create dynamic and attractive spaces to citizens. For this, information technology can (and should) help in creation of more dynamic environments. Augmented Reality is a technology that overlays virtual objects in the real world, and can be used for this purpose. One way to accomplish Augmented Reality is 3D reconstruction of the environment. This work presents a comparison of libraries for 3D reconstruction, with the objective of developing an application of RA using 3D reconstruction in the context of Creative Cities. In addition, this study also compares and analyzes the performance of virtualization tools (KVM and Virtual Box) when executing this type of application in a cloud infrastructure.**

***Keywords*— Cloud Computing, creative cities, 3D reconstruction, virtualization, performance evaluation.**

I. INTRODUÇÃO

DURANTE os primórdios do Movimento **Cidade Criativa** na década de 80, as ideias-base deste fenômeno estavam limitadas a temas relacionados a temáticas de arte e cultura. A concentração de ideias em torno dessas temáticas é justificável, uma vez que o grande esforço oriundo deste movimento advinha da classe artística; em uma tentativa de enaltecer o valor econômico do seu trabalho. O desenrolar deste movimento desencadeou diversos estudos e pesquisas sobre indústrias culturais, ambientes criativos e economia criativa, ampliando o conceito de Cidade Criativa para além de atividades artísticas. Hoje, enfatiza-se a busca por soluções criativas para questões ambientais ou modos inventivos para o convívio das pessoas. De acordo com [1], "*a Cidade Criativa é, portanto, uma mensagem clara para estimular a abertura mental, a imaginação e a participação pública*".

As tecnologias de informação e comunicação (TIC) não só podem, como devem auxiliar e fazer parte do desenvolvimento e da vivência deste conceito de Cidade Criativa; modificando a forma de enxergar a cidade e possibilitando a implementação de soluções criativas. A ideia é que as TICs possam nutrir a criatividade através de dados, para que os mesmos sejam convertidos em conhecimento e inovação [2]. As iniciativas de código aberto e de cocriação são indicativos de que é possível criar um ambiente criativo, no qual as pessoas são estimuladas a participar, comunicar e compartilhar suas ideias e experiências. As TICs podem ser utilizadas para criar ambientes criativos em cidades. Uma delas é a **realidade aumentada** (RA). Com essa tecnologia pode-se inserir elementos virtuais ao ambiente em tempo real [3]. Além de aplicações de entretenimento ([4] e [5]), RA também pode ser utilizada em diversas outras áreas, tais como medicina [6], setor de automóveis [7] e educação. Através de uma interface de comunicação e de interatividade próxima a linguagem natural, as tecnologias de RA são consideradas de fácil adaptação tanto para os usuários novos quanto para os experientes [8].

Existem várias técnicas para o desenvolvimento de apli- cações de RA. A mais comum delas é utilizando **marcadores**, que são objetos reais conhecidos utilizados como referência para sobreposição do objeto virtual [9]. Porém, essa técnica precisa do conhecimento prévio do marcador que estará pre- sente na cena, e inviabiliza, portanto, a utilização da mesma em aplicações que não possuem esta informação.

Para solucionar este problema dos marcadores, existem outras técnicas, como a baseada em **reconstrução 3D** ([10] e [11]). Esta técnica permite que, a partir de imagens de uma cena, seja construído um modelo tridimensional da mesma. Com base nas informações obtidas a partir deste modelo, é possível projetar os objetos virtuais, mesmo sem a presença de marcadores. Contudo, o desenvolvimento de aplicações de reconstrução 3D pode ser algo complexo e demorado. Essa complexidade vem da quantidade de cálculos de álgebra linear, geometria epipolar, geometria projetiva e probabilidade e estatística que são necessárias nos algoritmos de reconstrução 3D ([12], [13] e [14]); além de otimizações que são necessárias para o refinamento dos resultados da reconstrução, para que a saída seja de qualidade. Atualmente, existem diversas bibliote- cas que provêem esses cálculos e oferecem uma saída que pode ser utilizada para construir um sistema de RA, como libviso2 [15], libmv [16], Bundler [17] e FastCV [18].

Porém, apesar das bibliotecas auxiliarem no desenvolvi- mento das aplicações, os cálculos necessários para a recon- strução de ambientes 3D requerem bastante processamento [13], tornando inviável utilizar os *hardwares* de dispositivos móveis para esse processamento.ibilidade de gerenciamento de recurso, a infraestrutura da nuvem geralmente é virtualizada, e portanto, baseada em alocação dinâmica de máquinas virtuais [21]. Atualmente, há diversas tecnologias de virtualização, como virtualização total, virtualização assistida por *hardware* e para-virtualização [22], e portanto, é importante compreender de que forma esses tipos de virtualização podem impactar na execução de aplicações nas máquinas virtuais da nuvem.

Portanto, considerando estas problemáticas explicitadas an- teriormente, este trabalho se propõe a: (a) realizar um estudo acerca das bibliotecas mais relevantes e disponíveis para re- construção 3D; e (b) analisar o desempenho de uma ferramenta de reconstrução 3D em ambientes virtualizados com o objetivo de verificar sua viabilidade no ambiente de nuvem.

Este artigo está estruturado da seguinte forma: a Seção II trata do embasamento teórico acerca de reconstrução 3D; a Seção III descreve algumas ferramentas, bem como realiza uma avaliação qualitativa das mesmas, afim de selecionar uma pra os experimentos quantitativos; a Seção IV apresenta a descrição dos

G. Leoni, Universidade de Pernambuco (UPE), Caruaru, Pernambuco, Brasil, gutoleoni89@gmail.com

D. W. Beserra, Universidade Federal de Sergipe (UFS), Aracaju, Sergipe, Brasil, dw.beserra@gmail.com

P. T. Endo, Universidade de Pernambuco (UPE), Caruaru, Pernambuco, Brasil, patricia.endo@upe.br

experimentos e seus resultados são analisados na Seção V; os trabalhos relacionados da literatura são apre- sentados na Seção VI; e as conclusões e trabalhos futuros são delineados na Seção VII.

II. RECONSTRUÇÃO 3D

Muitas das aplicações de RA hoje em dia são baseadas em **marcadores**, que são elementos artificiais presentes na cena e que servirão como referência para a projeção dos objetos virtuais. Os marcadores precisam ser únicos na cena, pois a partir de sua localização será realizado o cálculo para se determinar as coordenadas do objeto que será projetado [9].

A Fig. 1 mostra um exemplo de marcador utilizado em uma aplicação de RA, neste caso a projeção de um prédio virtual.

Figura 1. Exemplo de marcador num cenário [23].

A projeção do objeto virtual pode ser vista na Fig. 2.

Figura 2. Demonstração de um objeto projetado sobre o marcador [23].

Contudo, para solucionar o problema do conhecimento prévio dos marcadores, existem outras técnicas, como a baseada em **reconstrução 3D** [10]. Como exemplo de in- tervenção cultural e tecnológica com reconstrução 3D, pode-se citar o projeto Re+Public [24] que surgiu com o objetivo principal de reimaginar o espaço público. Para isso, o Re+Public utiliza aplicações de tecnologia de imersão, RA, em dispositivos móveis para sobrepor conteúdo digital nas paredes e nos prédios em centros urbanos, como mostra a Fig. 3.

Porém, de acordo com [13], os cálculos necessários para a reconstrução de ambientes 3D têm um custo elevado, o que ocasiona a necessidade de alto processamento, tornando inviável utilizar os *hardwares* dos dispositivos móveis atuais. Tendo em vista essa limitação de capacidade de *hardware*, pode-se desenvolver aplicações de RA que utilizem servidores na nuvem, para que os mesmos realizem o processamento remoto da aplicação com reconstrução 3D. Para a realização da reconstrução 3D, existem alguns passos a serem seguidos, que são estruturados em um *pipeline* definido em [14]. Esses passos serão descritos a seguir.

Figura 3. Aplicação de reconstrução 3D para realizar intervenções em ambientes públicos [24].

A. Pipeline para reconstrução 3D

Nessa Seção, serão apresentados os passos do *pipeline* para realizar uma reconstrução 3D. Neste processo, é utilizado um rastreador que se baseia na distância da câmera que capturou as imagens de entrada. Esta distância pode ser chamada de *baseline*. As imagens de entrada podem ser *wide baseline*. Imagens classificadas assim são fotografias que representam uma determinada cena a partir de vários ângulos diferentes, sem necessidade de coerência temporal. A entrada pode ser composta também por imagens *short baseline*, que são ima- gens de câmeras próximas entre si que apresentam coerência temporal, como num vídeo [14]. Esse tipo de conjunto de imagens permitem um rastreamento mais eficiente.

Uma vez definido o conjunto de entrada para o sistema, pode-se seguir os passos do *pipeline* de reconstrução 3D. O trabalho apresentado em [14] define uma metodologia para a realização de reconstrução 3D baseada em imagens, como mostra a Fig. 4. Para o desenvolvimento dessa metodologia, foi realizado um estudo detalhado sobre os principais algorit- mos de reconstrução 3D.

Figura 4. *Pipeline* de reconstrução 3D de acordo com [14].

Este *pipeline* está dividido nos seguintes passos:

Execução do rastreador: o rastreador, no sistema de re- construção 3D, irá localizar as correspondências de um pixel entre uma imagem e as imagens posteriores a ela. A escolha do rastreador dependerá do tipo de entrada.

Escolha do *keyframe*: um *keyframe* é um quadro que possui baseline suficiente para ser feita a reconstrução 3D. Com ele é possível determinar a matriz fundamental [12] a partir de dois *keyframes*. O *keyframe* deve ser distante o suficiente do último *keyframe* para que seja possível fazer a triangulação dos pontos.

Execução do algoritmo SFM: *structure from motion* (SFM) é uma técnica que permite obter cenas 3D a partir de pontos de imagens 2D [25], e é um ponto importante para a reconstrução 3D baseada

em imagens. Existem vários algoritmos que podem calcular as matrizes fundamentais e essenciais [12], e estes podem se basear em algoritmos clássicos, como o *eight-point* [12] ou o *five-point* [26].

Refinamento da matriz essencial: é importante melhorar os resultados produzidos pelos algoritmos SFM. Deve ser feito o refinamento da matriz essencial [12] para ser extraída a pose da câmera em cada *frame*. Os refinamentos são feitos através da remoção dos *outliers* e de otimizações não-lineares.

Triangulação e filtragem dos pontos 3D: com as poses das câmeras estimadas, é possível fazer a triangulação para estimar o ponto 3D. Após a triangulação, pode ainda haver um refinamento das coordenadas 3D, ou a aplicação do algoritmo *bundle adjustment* [27] nos pontos 3D e nas poses das câmeras. Como os pontos 3D são calculados a partir de dados com ruídos, eles podem representar pontos com erros de reprojeção altos, sendo assim extremamente importante o refinamento.

Estimativa da pose da câmera: a escolha do algoritmo da pose da câmera depende da aplicação a ser desenvolvida, pois ele pode ser em tempo real ou *offline*.

Refinamento da pose da câmera: a pose estimada na etapa anterior ainda não é precisa para que seja feita a triangulação para se obter novos pontos 3D. Deve ser feito o refinamento da pose estimada, e uma das técnicas que pode ser utilizada é o *bundle adjustment* novamente.

III. FERRAMENTAS PARA RECONSTRUÇÃO 3D

A construção de uma aplicação que realize a reconstrução 3D, seja ela em tempo real ou *offline*, é algo complexo e trabalhoso [14]. Para ajudar na construção de sistemas como esses, existem algumas bibliotecas que podem facilitar bastante o desenvolvimento. Em [28], são apresentadas várias ferramentas para o auxílio da reconstrução 3D. Dentre elas algumas *open source*, algumas para Matlab [29] e outras pagas. Entre as ferramentas gratuitas, existem o OpenCV [30], openMGV [31], Bundler [17], libmv [16] e libviso2 [15].

Porém, decidiu-se reduzir o escopo deste trabalho para avaliação das ferramentas *open source* mais comumente utilizadas na literatura ([32], [33], [34], [35], [36], e [37]). Estas ferramentas serão descritas com mais detalhes a seguir e serão avaliadas de acordo com os passos implementados do *pipeline* proposto em [14].

A. libmv

A biblioteca libmv (Multiview library) [16] é uma biblioteca de reconstrução 3D que se baseia em imagens. De acordo com [38], a libmv é *"um sistema de código aberto projetado para unificar a comunidade de reconstrução em torno de uma base de código comum, extensível, e de alta qualidade"*.

O intuito desta biblioteca é ser integrada com o *software* para modelagem tridimensional, Blender [39]. De acordo com a sua documentação [16], a reconstrução densa está fora do escopo da libmv. A biblioteca conta com pouca documentação em seu site; há apenas uma breve descrição da mesma, falando sobre a sua proposta e o *link* para *download*. Não há documentação sobre o processo de instalação e dependências externas necessárias para a execução da biblioteca.

Devido à falta de suporte, foi inviável a configuração da biblioteca para testes. Com os arquivos baixados do repositório mais atualizado disponibilizado para *download*, não foi pos- sível construir o projeto da libmv. Um arquivo de configuração do pacote baixado possuía erro que impossibilitou a construção do projeto, inviabilizando a análise de desempenho da bib- lioteca.

B. libiviso2

A libviso2 [15] é uma biblioteca *cross-plataform* para a linguagem C++ para odometria visual baseada em câmera monocular e *stereo*. A odometria é uma técnica utilizada para estimar movimento de um corpo a partir de alguma base de dados. A odometria pode ser baseada em sensores [40] ou, como na libviso2, em câmeras para estimar o movimento.

A biblioteca utiliza, para câmeras monoculares, o algoritmo *eight-point* para a estimativa da matriz fundamental. Para mais informações sobre o algoritmo *eight-point* vide [12]. A odometria com câmeras monoculares da libiviso2 ainda está em desenvolvimento, por isso leva em consideração que a câmera estará se movendo a uma altura conhecida e fixa do solo, para que possa ser realizada a estimativa da escala.

Para a reconstrução 3D, ela utiliza o SFM (algoritmo *eight-point*). Ela utiliza o RANSAC ([41], [42], [43]) para a estimativa da matriz fundamental, mencionada na II-A. Uma vez estimada a matriz fundamental, a libviso2 calcula a matriz essencial. E com essas matrizes, ela calcula a rotação e a translação necessárias para a formação da pose da câmera.

A entrada para a biblioteca é um conjunto de imagens, e os parâmetros intrínsecos e extrínsecos da câmera. Os parâmetros intrínsecos relacionam as coordenadas dos pixels das imagens aos pontos do espaço medidos no sistema formado pela origem no centro da câmera. Os parâmetros extrínsecos se referem à posição e a orientação da câmera com relação a outro sistema de coordenadas [44]. A saída da biblioteca, para a reconstrução 3D com câmeras monoculares, é a pose da câmera, que é uma matriz 3x4 com a rotação e translação da câmera em cada *frame* do vídeo.

A biblioteca mostra ser uma boa escolha para quem precisa de uma ferramenta para agilizar o desenvolvimento de um sistema de reconstrução 3D. A biblioteca possui com uma boa documentação e suporte para a instalação e configuração [15]. Porém, a libviso2 precisa de refinamentos para prover um sistema de reconstrução 3D preciso. Isso se deve ao fato que existem ruídos nas capturas das imagens e são adicionados mais ruídos a cada fase do *pipeline*, através da solução de sistemas lineares e não lineares. Estes ruídos precisam ser minimizados através de um processo de refinamento. Assim, a biblioteca pode ser utilizada na situação onde não se deseja implementar todos os passos do *pipeline* de reconstrução.

Porém, precisa de refinamentos para se obter um resultado mais preciso.

C. Bundler

O Bundler [17] é uma biblioteca escrita em C/C++ que realiza o SFM tendo como entrada uma sequência de imagens, podendo ser *wide baseline*. A biblioteca recebe como entrada um conjunto de imagens, as *features* e as correspondências das imagens, e calcula a reconstrução 3D do ambiente e a geometria da cena como saída.

Ela está disponível para os sistemas operacionais Windows, Mac OS X e Linux. Bundler possui uma boa documentação [45], que inclui um manual contendo a descrição da biblioteca, explicações de como executar um exemplo e descrição da saída da biblioteca. Os códigos são bem comentados, descrevendo cada passo que será executado.

A biblioteca mostra ser bastante poderosa, implementando vários algoritmos necessários para a reconstrução 3D. Dentre eles estão a estimativa da matriz fundamental, o SFM, o *bundle adjustment*, a triangularização dos pontos, os cálculos de geometria epipolar,

entre outros. O Bundler é ser uma boa opção para construção de um sistema de reconstrução 3D, por ser uma biblioteca robusta e por implementar boa parte dos passos do *pipeline* e algoritmos complexos, agilizando e facilitando bastante o desenvolvimento.

A biblioteca não possui um rastreador de *features*, porém a documentação recomenda a utilização do SIFT [46] para a detecção de *keypoints* das imagens. *FastCV*

A fastCV [18] é uma biblioteca de visão computacional desenvolvida pela Qualcomm. De acordo com a API disponibilizada, a biblioteca fornece duas ferramentas principais para os desenvolvedores:

- Uma biblioteca de funções de visão computacional otimizada para funcionar de forma eficiente em dispos- itivos móveis; e
- Uma API de aceleração de *hardware*, independente do processador do dispositivo onde será executada.

A FastCV está disponível para sistemas operacionais Windows, Mac OS e Linux. Ela é compatível com os sistemas móveis Android e Windows Phone, e tem compatibilidade com os *chipsets* ARM e Snapdragon.

A biblioteca possui uma ótima documentação [47], que fornece toda a descrição da API, como os módulos que fornecem funcionalidades para visão computacional, processamento de imagens e RA. Entre os módulos, está o de reconstrução 3D. Nele estão descritos os métodos, suas entradas e saídas e o que eles executam.

Contudo, a FastCV não implementa todos os passos do *pipeline* descrito na Seção II-A. Ela possui métodos para o refinamento do posicionamento da câmera e cálculos da homografia (para informações sobre homografia, ver [12]).

A biblioteca fornece funções de alta performance para desenvolvimento de aplicações de reconstrução 3D. Porém, ela implementa apenas o último passo do *pipeline*; ela disponibi- liza funções que podem ajudar a implementar o primeiro passo, o que torna o desenvolvimento mais complexo e demorado.

Com relação ao *pipeline* da reconstrução 3D, a FastCV fornece funções que podem auxiliar, porém deixa a implementação de muitos passos nas mãos do desenvolvedor.

D. Análise das ferramentas

As ferramentas apresentadas serão analisadas sob dois aspectos. O primeiro é referente a quantidade de passos do *pipeline* de reconstrução 3D implementado por cada ferramenta; e o segundo aspecto é referente a características técnicas de desenvolvimento e uso de cada ferramenta.

A Tabela I exibe quais passos do *pipeline* são implementados por cada uma das ferramentas analisadas. Os dados foram retirados da documentação de cada biblioteca.

TABELA I
PASSOS IMPLEMENTADOS POR CADA BIBLIOTECA

Passos do *pipeline*	**libmv**	**libviso2**	**bundler**	**FastCV**
Definição do rastreador		X		
Escolha do *keyframe*	X			
Definição do algoritmo SFM	X	X	X	
Refinamento da matriz essencisal	X	X		
Triangulação e filtragem dos pontos	X	X	X	
Estimativa da pose da câmera	X	X	X	
Refinamento da pose da câmera			X	X

De acordo com a Tabela I, a ferramenta libviso2 implementa vários passos do *pipeline*, porém não implementa a escolha do *keyframe*. Isso implica que não é possível calcular a triangulação dos pontos, como descrito na Seção II-A. Mesmo que ela implemente a triangulação dos pontos, como a libviso2 não define os *keyframes*, a pose gerada por ela não é precisa. A libmv necessita de um rastreador para fornecer as cor- respondências entre as imagens, diferente da libviso2. Ela implementa quase todos os passos do *pipeline*, porém, assim como a libviso2, a libmv não refina a pose da câmera, logo as poses geradas por ela também não são precisas.

Já a ferramenta Bundler não possui rastreador, não faz o refinamento da matriz essencial e não define os *keyframes*. Porém, ela executa o refinamento da pose da câmera, diferente da libviso2 e da libmv. Como ela não executa o rastreamento, parte importante do *pipeline*, a documentação do Bundler recomenda a utilização de outra ferramenta para realizar essa função, o SIFT, como dito anteriormente na Sub-seção III-C. Enquanto as outras ferramentas implementam pelo menos quatro passos do *pipeline*, a FastCV só implementa o último passo. Logo ela não se mostra como uma ferramenta que auxilie a construção rápida de uma aplicação de reconstrução

3D, pois implementa apenas um passo do *pipeline*.

A Tabela II mostra algumas características de cada ferra- menta.

TABELA II
COMPARAÇÃO ENTRE AS BIBLIOTECAS

	Ferramentas			
Características técnicas	**libmv**	**libviso2**	**Bundler**	**FastCV**
Sistemas operacionais compatíveis	*	W e L	W, L e M	W, L e M
API disponível	não	não	não	sim
Modularizada	sim	sim	sim	sim
Linguagem de programação	C++	C++	C/C++	*

Como mostra a Tabela II, quase todas as ferramentas estão disponíveis para os principais sistemas operacionais; exceto a libviso2, que não está disponível para Mac OS. Não foi pos- sível encontrar na documentação da libmv em quais sistemas operacionais ela pode ser executada.

Entre as ferramentas analisadas, a única que possui API disponível é a FastCV; isto facilita o seu uso, pois apresenta os métodos disponíveis, a definição de suas entradas e saídas e uma breve descrição de cada um. Um fato importante é que todas as ferramentas são modularizadas, facilitando a utilização, pois o código é organizado. A libmv, a libviso2 são escritas em C++, enquanto Bundler é escrito em C e C++. A FastCV não disponibiliza em sua documentação qual é a linguagem de programação utilizada.

Dentre as ferramentas analisadas, todas trabalham com vídeo, e poderiam ser utilizadas. Porém a que atende melhor às necessidades deste trabalho é a ferramenta **Bundler**. Mesmo que ela não possua o rastreador, pode-se utilizado a ferramenta SIFT para realizar essa função. Seu código é bem documentado e modularizado, o que facilita o seu uso. Ela realiza diversos passos importantes do *pipeline* de reconstrução 3D e a sua saída é de qualidade: uma nuvem de pontos 3D e as poses das câmeras.

IV. DESIGN DOS EXPERIMENTOS

Os principais objetivos desta avaliação de desempenho são:

(a) determinar o *overhead* da ferramenta Bundler em ambientes Para tanto, os ambientes virtualizados para comparação foram Virtual Box [48] e *Kernel based Virtual Machine* (KVM) [49], usando o ambiente não-virtualizado como *baseline*. O Virtual Box e o KVM são sistemas híbridos com suporte a virtualização assistida por *hardware* e virtualização total. Estes virtualizadores

foram escolhidos porque oferecem uma camada de virtualização com poucas modificações no sistema operacional hospedeiro, um melhor isolamento de recursos.

Para alcançar os objetivos definidos neste trabalho, adaptou-se a metodologia utilizada por [50] para executar os experimentos, baseado em quatro principais atividades (Fig. 5): a primeira atividade é o planejamento e configuração dos experimentos, que definirá como os mesmos deverão ser conduzidos para se coletar as métricas. Nesta atividade, também será decidido o conjunto de imagens a ser utilizada. A atividade de execução dos experimentos é descrita na Fig. 6. A atividade de análise de desempenho aplica métodos estatísticos para prover acurácia nas informações provenientes dos experimentos.

Figura 5. Metodologia de execução dos experimentos (adaptado de [50]).

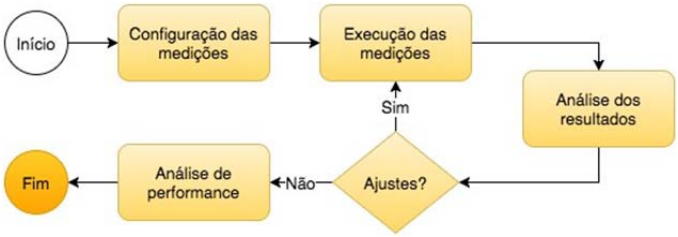

Figura 6. Atividades de execução dos experimentos.

Para a escolha do conjunto de imagens, decidiu-se utilizar imagens próximas ao cenário de uma cidade, visto que o contexto é aplicação para Cidades Criativas (Fig. 7). O conjunto de imagens foi coletado da base disponível pelo Karlsruhe Dataset [51]. O projeto disponibiliza sequências de imagens de alta qualidade gravadas a partir de um veículo em movimento.

A. *Infraestrutura de hardware e de software*

Os experimentos foram executados numa máquina HP Compaq 6000 Pro Microtower, equipado com processador virtualizados; e (b) verificar qual recurso é mais crucial no ambiente virtualizado, CPU, memória RAM e/ou tempo de processamento. Intel® Core 2 Duo E4800, operando com 3.00 GHz. Nos experimentos na máquina física foram usados 4 GB DDR3 de memória RAM. Os experimentos nas máquinas virtuais, foram utilizados 8 GB DDR3, porém apenas 4 GB foram usados nas máquinas virtuais. O sistema operacional utilizado foi o Lubuntu 15.10 LTS 64 bits.

Figura 7. Exemplo de imagem utilizada para os experimentos.

B. *Métricas*

Para a avaliação da ferramenta Bundler utilizando Virtual Box e KVM, definiu-se as seguintes métricas:

- Consumo médio de CPU;
- Consumo médio de memória RAM; e
- Tempo médio de processamento.

Todos os experimentos foram executados 32 vezes. As medições com os valores maiores e menores foram excluídas como *outliers*, e então calculou-se a média com as 30 medições restantes, com uma confiabilidade de 95%. Com o intuito de aferir o consumo das máquinas virtuais, foram realizadas medições indiretas do desempenho das mesmas.

V. RESULTADOS

A Tabela III apresenta o tempo necessário para se processar conjuntos com quantidades diferentes de imagens (5, 9, 13 e 130) utilizando a infraestrutura de *hardware* apresentada na Subseção IV-A, sem virtualização.

Com 5 imagens, o processamento é relativamente rápido, apenas 14s. Com um conjunto de 9 e 13 imagens, o proces- samento atinge 28s e 43s, respectivamente. Porém, com 130 imagens, o Bundler demorou aproximadamente 34min para fazer a reconstrução, o que prova que este tipo de aplicação consome bastante recurso computacional.

Levando em consideração que a aplicação trabalha com vídeo, o Bundler facilmente receberá uma grande quantidade de imagens, cujo processamento será bastante demorado. Tendo em vista a limitação dos dispositivos móveis com relação a processamento, torna-se necessário levar a aplicação para ser executada na nuvem para viabilizar a aplicação com bom tempo de processamento.

TABELA III
TEMPO DE EXECUÇÃO DA APLICAÇÃO BUNDLER NO AMBIENTE NÃO-VIRTUALIZADO

Quantidade de imagens	Tempo de execução (em minutos)
5	00:14
9	00:28
13	00:43
130	34:13

Como dito anteriormente, o objetivo dos experimentos apresentados neste trabalho é analisar o **impacto da virtualização** no processamento de aplicações 3D. A seguir, são apresentados os resultados referentes aos experimentos descritos na Seção IV.

A Tabela IV mostra os tempos de execução do Bundler no KVM e Virtual Box. O experimento durou mais tempo nos ambientes virtualizados, mostrando de certa forma o impacto da virtualização. Entre os virtualizadores, o KVM teve um tempo de execução menor que o Virtual Box.

TABELA IV
TEMPO DE EXECUÇÃO DOS EXPERIMENTOS NOS VIRTUALIZADORES KVM E VIRTUAL BOX

Quantidade de imagens	**Tempo de execução no KVM (em minutos)**	**Tempo de execução no Virtual Box (em minutos)**
5	00:15	00:19
9	00:30	00:36
13	00:46	00:55
130	34:49	37:27

A Fig. 8 mostra o bloxpot do consumo médio de memória RAM enquanto os experimentos são executados. O consumo médio da máquina física é próximo à 1400 MB. Enquanto o consumo médio da máquina física mais o virtualizador KVM é, em média, entre 2100 e 2200 MB; o que significa na prática um aumento máximo de 57% no consumo médio em relação ao ambiente não-virtualizado.

Com o Virtual Box executando na máquina física, o consumo médio varia entre 2700 e 3000 MB; implicando em um aumento máximo de 114% no consumo médio em relação ao ambiente não-virtualizado.

Verifica-se, portanto, que a utilização de virtualização aumenta substancialmente o uso de recursos de memória RAM para a execução da aplicação. É possível concluir também que o Virtual Box é mais oneroso que o KVM em termos de consumo de recursos de memória RAM, consumindo em média 800 MB a mais de memória (o dobro do registrado pelo KVM). De acordo a literatura, além de consumir mais memória, o Virtual Box também apresenta um pior desempenho em operações de leitura e escrita na mesma [22].

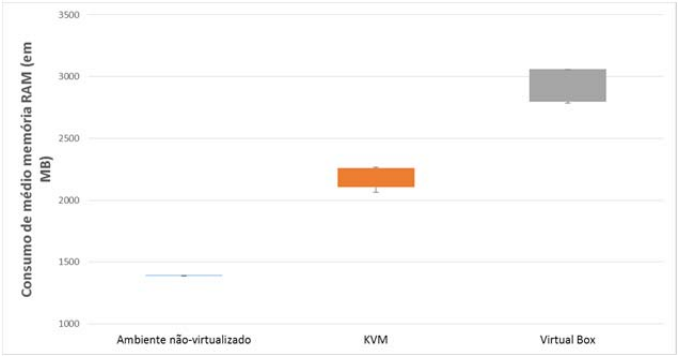

Figura 8. Consumo médio de memória RAM.

A Fig. 9 mostra o boxplot do consumo médio de CPU durante os experimentos. Em todos os ambientes testados, o consumo de CPU e o desempenho obtido foram similares, com diferenças percentuais abaixo de 5% para todos os casos. O consumo de CPU esteve abaixo de 49% tanto no ambiente não-virtualizado, quanto no virtualizado com o KVM; e acima de 50% (com picos de utilização de 52,1%) no ambiente virtualizado com o Virtual Box.

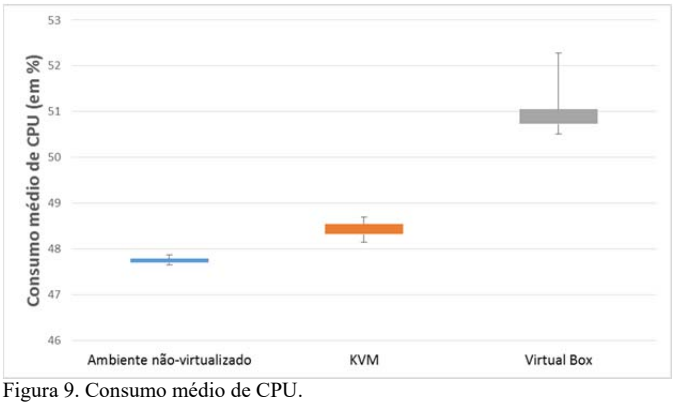

Figura 9. Consumo médio de CPU.

A Fig. 10 mostra o consumo de memória RAM durante a execução de um único experimento. É possível perceber oscilações no consumo de memória até os primeiros 27 segundos. Isso se deve ao fato de, nesse período de tempo, o SIFT estar em processo de leitura e processamento de imagens para a geração das *features* necessárias no processo de reconstrução 3D. Após essa etapa, as imagens são retiradas da memória, o consumo de memória é reduzido, e as oscilações deixam de ocorrer.

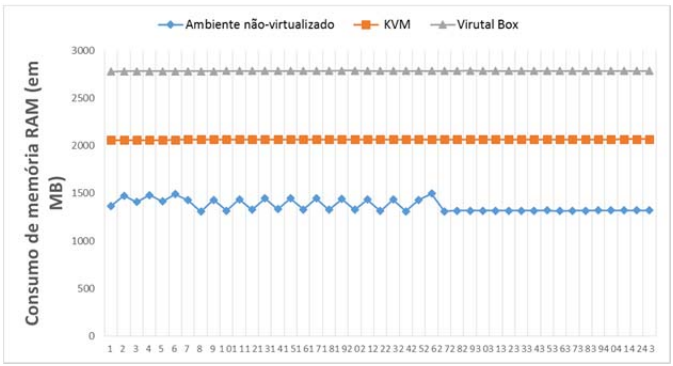

Figura 10. Consumo de memória RAM durante um experimento.

Já a Fig. 11 mostra o consumo de CPU (em %) durante a execução de um único experimento. Foi observado que o consumo médio de CPU durante o início do experimento oscila entre 45% e 50%. Percebe-se também que em todos os ambientes há um instante de tempo no qual o uso dos recursos de processamento decresce momentaneamente. No ambiente não-virtualizado e no virtualizado com o KVM, isso ocorre por volta dos 25 segundos, e no ambiente virtualizado com o Vir- tual Box, por volta dos 33 segundos. Essa anomalia coincide com o momento no qual o SIFT finaliza seu processamento e o Bundler começa a executar o algoritmo de SFM. Após o início da execução do SFM, a média de uso de recursos de CPU volta aos patamares anteriormente verificados. Conclui- se então, que o tempo de espera entre o fim da execução do SFIT e o do início do SFM é irrisório e que não há ociosidade no ambiente devido a essa transição.

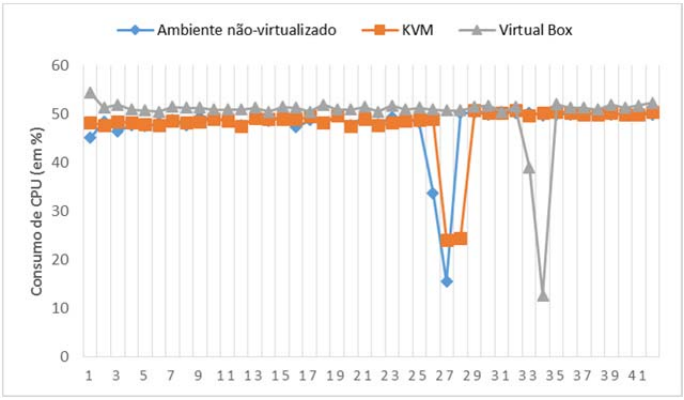

Figura 11. Consumo de CPU durante um experimento.

A. Discussão

Através dos resultados, observou-se que o consumo de CPU permanece constante no ambiente não-virtualizado, devido a constante necessidade de realização de cálculos intensivos de álgebra linear e geometria epipolar, necessários para a realização da reconstrução 3D. Por sua vez, o consumo de memória RAM varia em função da quantidade de imagens no experimento realizado. Isso ocorre devido a variação na resolução das imagens utilizadas pelo Bundler, e não devido ao uso de diferentes tecnologias de virtualização. No Bundler, as imagens são alocadas na memória, e portanto, quanto maior o número de *pixels*, mais espaço é necessário para sua manipulação das mesmas.

Considerando os experimentos com o Bundler nos ambi- entes virtuais, verificou-se que a execução dos experimentos demorou mais nos ambientes virtualizados, porém a sobre- carga no

desempenho não foi estatisticamente significante. Assim, pode-se considerar que todos os ambientes tiveram desempenho similar. O aumento do consumo de CPU devido a virtualização também pode ser considerado insignificante estatisticamente para esse tipo de aplicação. Já no tocante à utilização de memória RAM, percebeu-se um aumento significativo no consumo desse recurso, devido à utilização de virtualização (consequência de características intrínsecas à esse tipo de tecnologia).

De maneira geral, o KVM apresentou desempenho mais próximo ao ambiente não-virtualizado do que o Virtual Box. Isso ocorre devido a diferenças entre as tecnologias de virtualização utilizadas. Embora ambos sejam virtualizadores do Tipo 2, com um sistema operacional entre a camada de virtualização e o *hardware* [52], os seus mecanismos de acesso a memória principal são implementados de maneiras distintas. Essas diferenças de implementação implicam em diferentes desempenhos no acesso a memória principal, com o KVM sendo mais eficiente nesse aspecto que o Virtual Box. Essa diferença de desempenho no acesso a memória acaba sendo refletida no desempenho de todos os outros tipos de aplicação executados desses ambientes [22].

VI.TRABALHOS RELACIONADOS

Alguns trabalhos da literatura já propuseram a transferên- cia do processamento de aplicações dos *smartphones* para a nuvem.

Por exemplo, na área de saúde, os autores em [19] enviam imagens para a nuvem, e o retorno é o modelo 3D e as texturas do rosto para **reconstrução facial**. Como resultado, o tamanho do aplicativo instalado no *smartphone* foi reduzido de 33MB para 1,14MB. O tempo da reconstrução facial também diminuiu de 266ms para 204ms. Finalmente, o tempo total de execução foi reduzido de 1,9s para 1,4s.

Ainda na área de saúde, o trabalho desenvolvido em [20] avalia um sistema de reconstrução 3D a partir de imagens de **tomografias**, chamado Mangoose++, que é executado na nuvem. Nos testes realizados nas maquinas virtuais, as que conseguiram melhores resultados foram as que tinham mais núcleos no processador, como esperado. Isso se deve ao fato do Magoose++ implementar um algoritmo de reconstrução de forma otimizada e paralela.

Já no trabalho [53], é dada atenção para o gerenciamento e desenvolvimento de estratégias para **cidades inteligentes**, com o foco no meio ambiente e cultura. Foram feitos os modelos 3D dos locais utilizando *softwares open source* que usam técnicas de reconstrução 3D e fotogrametria. Foi realizada uma avaliação da qualidade do *software* de reconstrução 3D que utiliza o SFM e a reconstrução Multi-View densa.

Em [54], foi feita uma avaliação do desempenho de *softwares* de reconstrução 3D, e medido seus desempenhos na reconstrução de ambientes. Foram utilizados 4 *data sets* para testar o desempenho das ferramentas e foram analisados os modelos 3D gerados por elas.

Em [55], é apresentada uma aplicação de RA para dispositivos móveis. Ela apresenta um modelo digital de Roman Villa, na Espanha. Essa aplicação funciona com base em sensores dos *smartphones* e no GPS dos mesmos. O objetivo dessa aplicação é disponibilizar e armazenar a herança cultural presente nesse lugar histórico da Espanha, pois o mesmo não está totalmente disponível para todos.

O trabalho apresentado aqui difere destes trabalhos da literatura, pois os objetivos e o foco são diferentes. Em [19], a saída do sistema não pode ser utilizada em Cidades Criativas, pois tem foco em reconstrução 3D de faces. Enquanto em [20], a entrada do sistema são imagens de tomografias, e não imagens provenientes de câmeras ou de *smartphones*. Em [53] e [54], as aplicações foram testadas em máquinas físicas, e neste artigo, o objetivo é realizar a avalização do desempenho das aplicações em máquinas virtuais, como as que são usadas em servidores na nuvem. Já em [55], o cenário é Cidade Criativa, mas a aplicação desenvolvida contempla um lugar específico, que foi digitalizado e usado como objeto virtual.

Assim, diferente dos trabalhos da literatura, este artigo propõe uma análise das bibliotecas de reconstrução 3D que podem auxiliar no desenvolvimento de aplicações de RA no contexto de Cidades Criativas. Tais bibliotecas precisam receber como entrada um conjunto de imagens (*frames de um vídeo* ou de uma câmera) e ter como saída um modelo tridimensional do ambiente presente nas imagens. Além disso, este trabalho também realiza uma avaliação do **impacto da virtualização** na execução de uma aplicação de reconstrução 3D.

VII. CONCLUSÕES E TRABALHOS FUTUROS

Uma vez que desenvolver um sistema de reconstrução 3D é algo complexo, pode-se utilizar bibliotecas para ajudar no desenvolvimento de uma aplicação de reconstrução 3D com foco em RA. A quantidade de bibliotecas que promete fazer o processo completo de reconstrução 3D é bem pequena. Além de existirem poucas opções, as existentes não são tão eficientes.

Como o objetivo deste trabalho foi utilizar uma biblioteca para desenvolver uma aplicação de RA que utiliza reconstrução 3D, concluiu-se que, dentre as analisadas, **o Bundler foi a melhor opção**. Isso se deve ao fato de ela implementar mais passos necessários do *pipeline* de reconstrução 3D apresentado na seção II-A. Além disso sua saída se mostra de melhor qualidade para o cenário de avaliação em questão. Sua saída é composta pelas poses das câmeras e uma nuvem de pontos que formam o modelo tridimensional do que se está reconstruindo. Como as aplicações de reconstrução 3D demandam bastante poder computacional, uma alternativa para executá-las em tempo hábil - em uma perspectiva de usuário - é transferir o processamento para servidores na nuvem. Como tais servidores costumam executar as aplicações em máquinas virtuais, este trabalho realizou testes para verificar quais impactos poderiam ser causados pela utilização de virtualização no desempenho
da ferramenta Bundler.

De acordo com os dados coletados, a virtualização não impacta consideravelmente no desempenho, mas demanda maior uso de **memória RAM**. O consumo de memória no KVM foi de 57% a mais com relação ao ambiente não virtualizado, enquanto no Virtual Box foi de 114%. Como essa memória extra será oriunda do ambiente de nuvem, e não dos dispositivos moveis, então ainda sim é válido migrar o processamento da aplicação para a nuvem em uma perspectiva de desempenho. Além disso, também foi possível concluir que, no geral, **o KVM obteve melhor desempenho que o Virtual Box**, devido ao menor consumo de CPU, menor consumo de memória RAM e menor tempo de execução.

As principais limitações do trabalho foram com relação à infraestrutura utilizada para os experimentos, limitando assim a realização de testes mais próximos aos ambientes de nuvem presentes hoje no mercado; e com relação a quantidade de

virtualizados utilizados nos experimentos, podendo adicionar novos virtualizados, bem como novas tecnologias, como os *containers*.

Como foi visto neste trabalho, a técnica de virtualização não impacta consideravelmente no desempenho da aplicação no ambiente virtual. Mas, com o objetivo de aprofundar o entendimento sobre os principais virtualizadores existentes atualmente, pretende-se, como trabalho futuro, realizar experimentos com outras tecnologias de virtualização, como as soluções Docker e LXC (*Linux Container*), que são implementações promissoras de *containers* para virtualização de aplicações, que apresentam desempenho próximo do ambiente não-virtualizado [56].

Além disso, como trabalho futuro também pretende-se finalizar a aplicação de RA e avaliar métricas de percepção do usuário (QoE - *Quality of Experience*) ao se transferir o processamento da aplicação para a nuvem computacional. Pretende-se realizar avaliações qualitativas e quantitativas, usando o *delay* do tempo de resposta e acesso a disco (I/O) como métricas para novos experimentos, por exemplo.

AGRADECIMENTOS

Este trabalho foi realizado com apoio da Fundação de Amparo à Ciência e Tecnologia de Pernambuco (FACEPE), Edital 01/2015.

REFERÉNCIAS

[1] C. Landry and F. Bianchini, *The creative city*. Demos, 1995, vol. 12. [2] A. C. F. Reis, *Cidades criativas*. SESI SENAI Editora, 2012.

[3] C. Kirner and R. Siscoutto, "Realidade virtual e aumentada: conceitos, projeto e aplicações," in *Livro do IX Symposium on Virtual and Augmented Reality, Petrópolis (RJ), Porto Alegre: SBC*, 2007.

[4] "Hololens microsoft," acesso em 05 de novembro de 2015. [Online]. Disponível em: http://www.microsoft.com/microsoft-hololens/en-us. [5] "Ingress niantic labs," acesso em 05 de novembro de 2015. [Online]. Disponível em: https://www.ingress.com/.

[6] W. Birkfellner, M. Figl, K. Huber, F. Watzinger, F. Wanschitz, J. Hummel, R. Hanel, W. Greimel, P. Homolka, R. Ewers *et al.*, "A head- mounted operating binocular for augmented reality visualization in medicine-design and initial evaluation," *Medical Imaging, IEEE Trans- actions on*, vol. 21, no. 8, pp. 991–997, 2002.

[7] M. W. d. S. Ribeiro and E. R. Zorzal, "Realidade virtual e aumen- tada: Aplicações e tendências," *XIII Simpósio de Realidade Virtual e Aumentada, Uberlândia-MG-Brasil*, 2011.

[8] N. Costa Bastos, "Uma metodologia para avaliação de usabilidade de interfaces de realidade mista interativas," 2007.

[9] A. A. B. Buccioli, E. R. Zorzal, and C. Kirner, "Usando realidade virtual e aumentada na visualização da simulação de sistemas de automação industrial," in *SVR2006-VIII Symposium on Virtual Reality*, 2006.

[10] M.-O. Berger, "Resolving occlusion in augmented reality: a contour based approach without 3d reconstruction," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*. IEEE, 1997, pp. 91–96.

[11] Y. Tian, Y. Long, D. Xia, H. Yao, and J. Zhang, "Handling occlusions in augmented reality based on 3d reconstruction method," *Neurocom- puting*, vol. 156, pp. 96–104, 2015.

[12] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[13] M. Brown and D. G. Lowe, "Unsupervised 3d object recognition and reconstruction in unordered datasets," in *3-D Digital Imaging and Modeling, 2005. 3DIM 2005. Fifth International Conference on*. IEEE, 2005, pp. 56–63.

[14] T. S. M. C. de Farias, "Metodologia para reconstrução 3d baseada em imagens," Ph.D. dissertation, Centro de Informática, Universidade Federal de Pernambuco, Pernambuco, Brazil, 2012.

[15] "libviso2," acesso em 15 de novembro de 2015. [Online]. Disponível em: http://www.cvlibs.net/software/libviso/.

[16] "libmv," acesso em 15 de novembro de 2015. [Online]. Disponível em: https://developer.blender.org/project/profile/59/.

[17] "Bundler," acesso em 28 de janeiro de 2016. [Online]. Disponível em: http://www.cs.cornell.edu/ snavely/bundler/.

[18] "FastCV qualcomm," acesso em 28 de janeiro de 2016. [Online]. Disponível em: https://developer.qualcomm.com/software/fastcv-sdk. [19] P. H. Truong, C.-W. Park, M. Lee, T.-Y. Lee, and G.-M. Jeong, "3d facial model reconstruction on mobile devices using cloud," in *Information and Communication Technology Convergence (ICTC), 2014 International Conference on*. IEEE, 2014, pp. 943–944.

[20] E. Serrano, G. Bermejo, J. Garcia Blas, and J. Carretero, "Evaluation of the feasibility of making large-scale x-ray tomography reconstructions on clouds," in *Cluster, Cloud and Grid Computing (CCGrid), 2014 14th IEEE/ACM International Symposium on*. IEEE, 2014, pp. 748–754.

[21] H. Ruschel, M. S. Zanotto, and W. C. MOTA, "Computação em nuvem," *Pontifícia Universidade Católica do Paraná, Curitiba, Brazil*, 2010.

[22] D. Beserra, F. Oliveira, J. Araujo, F. Fernandes, A. Araújo, P. Endo, P. Maciel, and E. D. Moreno, "Performance evaluation of hypervisors for hpc applications," in *Systems, Man and Cybernetics, 2015 IEEE International Conference On*. IEEE, 2015, pp. 846–851.

[23] "Galileu visita maior realidade aumentada do mundo," acesso em 09 de junho de 2016. [Online]. Disponível em: http://goo.gl/1SRn.

[24] "Re+republic," http://www.republiclab.com, acesso em 03 de abril de 2016. [Online]. Disponível em:.

[25] M. Westoby, J. Brasington, N. Glasser, M. Hambrey, and J. Reynolds, "'structure-from-motion'photogrammetry: A low-cost, effective tool for geoscience applications," *Geomorphology*, vol. 179, pp. 300–314, 2012.

[26] D. Nistér, "An efficient solution to the five-point relative pose problem," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 6, pp. 756–770, 2004.

[27] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment—a modern synthesis," in *Vision algorithms: theory and practice*. Springer, 2000, pp. 298–372.

[28] A. KOUTSOUDIS, F. ARNAOUTOGLOU, and G. PAVLIDIS, "Di- garch 2015 notes structure from motion–multiple view stereovision (sfm/mvs) an introduction on how to perform sfm/mvs."

[29] "Matlab," acesso em 21 de abril de 2016. [Online]. Disponível em: http://goo.gl/ajP7GT.

[30] "Opengl," acesso em 02 de novembro de 2015. [Online]. Disponível em: https://www.opengl.org/.

[31] "Opengv," acesso em 21 de abril de 2016. [Online]. Disponível em: https://github.com/openMVG/openMVG.

[32] H. Silva, E. Silva, and A. Bernardino, "Probabilistic stereo egomotion transform," in *Robotics and Automation (ICRA), 2014 IEEE Interna- tional Conference on*. IEEE, 2014, pp. 5461–5466.

[33] J. d. L. Silveira, E. O. Freire, L. Molina, and E. A. Carvalho, "A simple visual odometry approach based on point clouds," in *2015 12th Latin American Robotics Symposium and 2015 3rd Brazilian Symposium on Robotics (LARS-SBR)*. IEEE, 2015, pp. 204–209.

[34] Z. Huang, J. Zhang, and Q. Zhang, "A hybrid method for video stabilization and retargeting," in *Proceedings of the Fifth International Conference on Internet Multimedia Computing and Service*. ACM, 2013, pp. 46–51.

[35] B. M. Chrzan, "Liveness detection for face recognition," Ph.D. disser- tation, Masarykova univerzita, Fakulta informatiky, 2014.

[36] G. Schindler and F. Dellaert, "Probabilistic temporal inference on reconstructed 3d scenes," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 1410–1417.

[37] J. Heller, M. Havlena, M. Jancosek, A. Torii, and T. Pajdla, "3d reconstruction from photographs by cmp sfm web service," in *Machine Vision Applications (MVA), 2015 14th IAPR International Conference on*. IEEE, 2015, pp. 30–34.

[38] K. Mierle, "Open source 3d reconstruction from video," Master's thesis, University of Toronto, 2008.

[39] "Blender," acesso em 02 de dezembro de 2015. [Online]. Disponível em: https://www.blender.org/.

[40] R. García-García, M. A. Sotelo, I. Parra, D. Fernández, J. E. Naranjo, and M. Gavilán, "3d visual odometry for road vehicles," *Journal of Intelligent and Robotic Systems*, vol. 51, no. 1, pp. 113–134, 2008.

[41] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[42] R. Raguram, J.-M. Frahm, and M. Pollefeys, "A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus," in *European Conference on Computer Vision*. Springer, 2008, pp. 500–513.

[43] X. Tong, Z. Ye, Y. Xu, S. Liu, L. Li, H. Xie, and T. Li, "A novel sub- pixel phase correlation method using singular value decomposition and unified random sample consensus," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 8, pp. 4143–4156, 2015.

[44] R. V. F. Meireles, "Interface com computador por controlo visual de cursores," 2002.

[45] "Bundler Manual," acesso em 28 de janeiro de 2016. [Online]. Disponível em: http://goo.gl/XTQsHI.

[46] "Sift keypoint detector," acesso em 17 de abril de 2016. [Online].

Disponível em: http://www.cs.ubc.ca/ lowe/keypoints/.
[47] "Fastcv public api documentation," acesso em 28 de janeiro de 2016. [Online]. Disponível em: https://developer.qualcomm.com/docs/fastcv/api/index.html.
[48] "Virtual box," acesso em 02 de maio de 2016. [Online]. Disponível em: https://www.virtualbox.org/.
[49] "Kvm," acesso em 02 de maio de 2016. [Online]. Disponível em: http://www.linux-kvm.org/page/Main_Page.
[50] E. Sousa, P. Maciel, E. Medeiros, D. Souza, F. Lins, and E. Tavares, "Evaluating eucalyptus virtual machine instance types: A study con- sidering distinct workload demand," in *CLOUD COMPUTING 2012, The Third International Conference on Cloud Computing, GRIDs, and Virtualization*, 2012, pp. 130–135.
[51] "Karlsruhe dataset," acesso em 02 de maio de 2016. [Online]. Disponível em: http://www.cvlibs.net/datasets/karlsruhe_sequences/. [52] M. Åsberg, N. Forsberg, T. Nolte, and S. Kato, "Towards real-time scheduling of virtual machines without kernel modifications," in *Emerging Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on*. IEEE, 2011, pp. 1–4.
[53] A. De Masi, "Smart cultural heritage and open source as regeneration of historical centers: Fruition, conservation and preservation in current methods of 2d/3d digitization and 3d modelling," in *Big Data and Cloud Computing (BdCloud), 2014 IEEE Fourth International Conference on*. IEEE, 2014, pp. 729–736.
[54] J. Schöning and G. Heidemann, "Evaluation of multi-view 3d reconstruction software," in *Computer Analysis of Images and Patterns*. Springer, 2015, pp. 450–461.
[55] J. L. Martínez, S. Álvarez, J. Finat-Saez, F. J. Delgado, and J. Finat, "Augmented reality to preserve hidden vestiges in historical cities. a case study," *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 40, no. 5, p. 61, 2015.
[56] D. Beserra, E. D. Moreno, P. Takako Endo, J. Barreto, D. Sadok, and S. Fernandes, "Performance analysis of lxc for hpc environments," in *Complex, Intelligent, and Software Intensive Systems (CISIS), 2015 Ninth International Conference on*. IEEE, 2015, pp. 358–363.

Guto Leoni Santos é graduando em Sistemas de Informação pela Universidade de Pernambuco (UPE) - Campus Caruaru. É bolsista de iniciação científica (PIBIC) pela Fundação de Amparo à Ciência e Tec- nologia de Pernambuco (FACEPE). Tem experiência na área de Ciência da Computação e Sistemas de Informação, com ênfase em desenvolvimento de sis- temas, atuando principalmente nos seguintes temas: reconstrução 3D, realidade aumentada e aplicativos para cidades inteligentes.

David Willians dos Santos Cavalcanti Beserra é mestrando em Ciência da Computação pela Uni- versidade Federal de Sergipe (UFS) e bacharel em Ciência da Computação pela Universidade Federal Rural de Pernambuco (UFRPE). Realiza pesquisas em Computação de Alto Desempenho, com foco em avaliação de desempenho de infraestruturas virtualizadas.

Patricia Takako Endo possui doutorado (2014) e mestrado (2008) em Ciência da Computação pela Universidade Federal de Pernambuco (UFPE) e grad- uação em Engenharia da Computação (2005) pela Universidade Federal do Pará (UFPA). É professora adjunta da Universidade de Pernambuco (UPE) desde 2010; e é pesquisadora do Grupo de Redes de Computadores e Telecomunicações (GPRT) desde 2009. Suas áreas de interesse atuais são: computação em nuvem, gerenciamento de recursos e cidades inteligentes e criativa.