Universidade Federal do Espírito Santo – UFES

Centro Tecnológico

Programa de Pós-Graduação em Informática

Ricardo Mendes Costa Segundo

Aplicando Crowdsourcing na Sincronização de Vídeos Gerados por Usuários

Vitória, ES
2017

Ricardo Mendes Costa Segundo

Aplicando Crowdsourcing na Sincronização de Vídeos Gerados por Usuários

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Informática da Universidade Federal do Espírito Santo, como requisito para a obtenção do Grau de Doutor em Ciência da Computação.

Orientador: Prof. Dr. Celso Alberto Saibel Santos

Vitória – ES
2017

Dados Internacionais de Catalogação-na-publicação (CIP)
(Biblioteca Setorial Tecnológica,
Universidade Federal do Espírito Santo, ES, Brasil)
Bibliotecária: Ilane Coutinho Duarte Lima – CRB-6 ES-000348/O

Costa Segundo, Ricardo Mendes, 1987-
C837a Aplicando crowdsourcing na sincronização de vídeos gerados por usuários / Ricardo Mendes Costa Segundo. – 2017.
168 f. : il.

Orientador: Celso Alberto Saibel Santos.
Tese (Doutorado em Ciência da Computação) – Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Sincronização. 2. Multimídia interativa. 3. Computação humana. 4. Sistemas operacionais distribuídos (Computadores). I. Santos, Celso Alberto Saibel. II. Universidade Federal do Espírito Santo. Centro Tecnológico. III. Título.

CDU: 004

Ricardo Mendes Costa Segundo

Aplicando Crowdsourcing na Sincronização de Vídeos Gerados por Usuários

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Informática da Universidade Federal do Espírito Santo, como requisito para a obtenção do Grau de Doutor em Ciência da Computação.

Trabalho aprovado em 30 de outubro de 2017:

_____

Prof. Dr. Celso Alberto Santos Saibel (Orientador)
Universidade Federal do Espírito Santo

_____

Prof. Dr. José Gonçalves Pereira Filho
Universidade Federal do Espírito Santo

_____

Prof. Dr. Rodrigo Laiola Guimarães
Universidade Federal do Espírito Santo

_____

Prof. Dr. Guido Lemos de Souza Filho
Universidade Federal da Paraíba

_____

Prof. Dr. Roberto Willrich
Universidade Federal de Santa Catarina

A meus pais, Maria Sueli Nunes Costa e Ricardo Mendes Costa, que me deram todas as ferramentas para trilhar meu caminho.

AGRADECIMENTOS

Abrir mão de algo na vida nunca é fácil. Juntando a isto o fato de ir para um lugar desconhecido, sem familiares e amigos e sem nunca ter colocado os pés lá. Foi assim que comecei esta jornada: abrindo mão de minha cidade natal com família e amigos para morar em Vitória, lugar que nunca tinha visto. Mas nunca só, sempre apoiado por minha família e amigos, não muitos, mas verdadeiros.

Penso que apesar de ter sido eu a vir para cá, a maior dificuldade foi para aqueles que ficaram. Maria Sueli Nunes Costa e Ricardo Mendes Costa, obrigado por todo apoio que me foi dado, por compreender e incentivar esta minha caminhada, suportar a distância e me permitir estas conquistas em minha vida. Obrigado Mainha e Painho.

A meu irmão, agradeço pelo exemplo que sempre tive. Do esforço que fez para alcançar seus sonhos. Do grande marido e pai que é. Obrigado Tom.

Tenho que agradecer a essa minha linda canela verde, que faz “pocar” meu coração. Que me mostra os caminhos para uma vida feliz. Obrigado meu amor, minha Juliana.

Aos vários amigos que tenho. A aqueles que deixei em João Pessoa, dos quais muitos também já seguiram em frente, inclusive para o outro lado do mundo, não é Sr. Marinho? E claro, aos novos que por estes lados me ajudaram a chamar este lugar de lar, ajudando nos tempos difíceis, e compartilhando a alegria dos bons momentos. Obrigado Robert, Mozer e Izon. E amigos que além de tudo isso também ajudaram na construção dessa tese diretamente, com horas de trabalho e muita conversa: obrigado Victor, Wancharle, Saulo e Jéssica. Em especial, obrigado Marcello Novaes, pois nesta reta final, a coisa teria sido muito mais difícil sem sua ajuda.

Impossível esquecer dos mestres. Primeiro ao professor Celso, que me acolheu como aluno e me ajudou muito, tanto como orientador, quanto como amigo. E apesar da distância, obrigado Tati, eterna orientadora e exemplo de como me portar com meus alunos.

Agradecimentos à Fundação de Amparo à Pesquisa e Inovação de Espírito Santo pelo apoio financeiro (processo 64362361).

“Happiness is not something ready made.
It comes from your own actions.” - (Dalai Lama)

RESUMO

Crowdsourcing é uma estratégia para resolução de problemas baseada na coleta de resultados parciais a partir das contribuições de indivíduos, agregando-as em um resultado unificado. Com base nesta estratégia, esta tese mostra como a *crowd* é capaz de sincronizar um conjunto de vídeos produzidos por usuários quaisquer, correlacionados a um mesmo evento social. Cada usuário filma o evento com seu ponto de vista e de acordo com suas limitações (ângulo do evento, oclusões na filmagem, qualidade da câmera utilizada, etc.). Nesse cenário, não é possível garantir que todos os conteúdos gerados possuam características homogêneas (instante de início e duração de captura, resolução, qualidade, etc.), dificultando o uso de um processo puramente automático de sincronização. Além disso, os vídeos gerados por usuário são disponibilizados de forma distribuída entre diversos servidores de conteúdo independentes. A hipótese desta tese é que a capacidade de adaptação da inteligência humana pode ser usada para processar um grupo de vídeos produzidos de forma descoordenada e distribuída, e relacionados a um mesmo evento social, gerando a sua sincronização. Para comprovar esta hipótese, as seguintes etapas foram executadas: (i) o desenvolvimento de um método de sincronização para múltiplos vídeos provenientes de fontes independentes; (ii) a execução de um mapeamento sistemático acerca do uso de *crowdsourcing* para processamento de vídeos; (iii) o desenvolvimento de técnicas para o uso da *crowd* na sincronização de vídeos; (iv) o desenvolvimento de um modelo funcional para desenvolvimento de aplicações de sincronização utilizando *crowdsourcing*, que pode ser estendido para aplicações de vídeos em geral; e (v) a realização de experimentos que permitem mostrar a capacidade da *crowd* para realizar a sincronização. Os resultados encontrados após estas etapas mostram que a *crowd* é capaz de participar do processo de sincronização e que diversos fatores podem influenciar na precisão dos resultados obtidos.

Palavras-chave: Vídeo. Vídeos Gerados pelo Usuário. Sincronização. Crowdsourcing.

ABSTRACT

Crowdsourcing is a solve-problem strategy based on collecting contributions with partial results from individuals and aggregating them into a major problem solution. Based on this strategy, this thesis shows how the crowd can synchronize a set of videos generated by users, correlated to an event. Each user captures the event with its personal viewpoint and according to its limitations. In this scenario, it is not possible to ensure that all generated contents have homogeneous characteristics (starting time and duration, resolution, quality, etc.), hindering the use of a purely automatic synchronization process. Additionally, user generated videos are distributed available between several independent content servers. The assumption of this thesis is that the ability of human intelligence to adapt can be used to render a group of videos produced in an uncoordinated and distributed manner generating its synchronization. To prove this hypothesis, the following steps were executed: (i) the development of a synchronization method for multiple videos from independent sources; (ii) The execution of a systematic mapping about the use of crowdsourcing for processing videos; (iii) The development of techniques for the use of the crowd in synchronizing videos; (iv) The development of a functional model for developing synchronization applications using crowdsourcing, which can be extended for general video applications; and (v) The execution of experiments to show the ability of the crowd to perform the synchronization. The results found show that the crowd can participate in the synchronization process and that several factors can influence the accuracy of the results obtained.

Keywords: Video. User Generated Video. Synchronization. Crowdsourcing.

SUMÁRIO

1. INTRODUÇÃO

Há alguns anos, o acesso aos vídeos estava limitado ao que era transmitido pela TV, conteúdos gravados e armazenados individualmente pelos usuários ou adquiridos em lojas ou em vídeo locadoras. O modelo bem estabelecido de consumo de conteúdo audiovisual pela TV (*Produce–Deliver–Consume*), onde uma estação de TV controla todo o conteúdo enviado a todos, era a regra. No entanto, hoje em dia, o fácil acesso aos meios de produção e divulgação de conteúdo digital criou um novo modelo, o ESC (*Edit-Share-Control*) (CESAR e CHORIANOPOULOS, 2009). De acordo com o modelo ESC, tanto usuários finais (consumidores) quanto emissoras podem participar do ciclo de produção de conteúdo audiovisual. Além disso, os usuários podem também consumir conteúdo de várias fontes, seja de repositórios Web ou de serviços de *streaming*.

Os vídeos tornaram-se uma parte importante da Internet, uma vez que representam uma grande porcentagem do seu tráfego atual e, até 2021, representarão 82% (CISCO, 2017). O acesso aos vídeos faz parte do cotidiano das pessoas, seja visualizando comentários em redes sociais, seja assistindo às filmagens geradas por outros usuários. Serviços de streaming de vídeo, como o Netflix, rivalizam com empresas de TV (ALILOUPOUR, 2016), enquanto plataformas como YouTube recebem horas de vídeos gerados por usuários a cada segundo (ALGUR e BHAT, 2016). Usando dispositivos móveis e a Internet, as pessoas são capazes de gerar seu próprio conteúdo, e compartilhar experiências e opiniões.

O processo de geração de vídeos pelos usuários possui características bastante particulares. Em geral, a produção dos chamados *UGVs* (do inglês User Generated Videos) (BANO e CAVALLARO, 2015) é feita de forma não coordenada e não planejada. Mesmo em eventos sociais nos quais os usuários compartilham um mesmo espaço durante um mesmo intervalo de tempo, cada usuário tende a produzir seu próprio conteúdo, a partir do seu ponto de vista, de forma independente, compartilhando o conteúdo gerado, também de forma independente, logo após a captura. Além disso, os conteúdos produzidos são heterogêneos, ou seja, possuem diferentes resoluções, qualidade, sentido, tamanho e formato, tornando seu processamento desafiador (SHRESTHA, WITH, *et al.*, 2010).

Pelo exposto, pode-se perceber que a disseminação de vídeos via Internet cria uma série de desafios para a área de computação. Um deles é o de sincronizar o conteúdo dos múltiplos vídeos, capturados em paralelo, por diferentes usuários, em um mesmo evento social. Nesta tese, um evento social, ou simplesmente evento, define algo que reúne um grupo de pessoas que compartilham um mesmo espaço durante um determinado intervalo de tempo. Assim, pode-se dizer que os conteúdos gerados por esse grupo de pessoas em um mesmo evento é fortemente correlacionado no tempo e no espaço.

A sincronização destes vídeos permitiria, por exemplo, recontar a história do evento social a partir dos múltiplos conteúdos capturados, ou ainda, ofereceria ao espectador diferentes ângulos de filmagem e/ou diferentes fontes de áudio para assistir a este evento. A anotação, a descrição, o agrupamento e a avaliação destes conteúdos são outros desafios interessantes relacionados.

Soluções para estes desafios costumam ter um custo de processamento elevado devido à quantidade de dados que deve ser processado e à heterogeneidade dos conteúdos vídeos. Desafios semelhantes para outros tipos de mídia, como imagens, tiveram sucesso com a aplicação de abordagens alternativas, nas quais pessoas são responsáveis pelo processamento e extração de informações das mídias. Neste caso, diz-se que o problema foi solucionado por meio de computação humana (LAW e AHN, 2011).

Um exemplo do tipo de abordagem anterior é o ESP Game (VON AHN, MAURER, *et al.*, 2008). A interação com o game permite descrever imagens a partir de contribuições humanas sob a forma de anotações significativas dos seus conteúdos, seguindo um modelo de produção *crowdsourcing*. Esse modelo de produção agrega diferentes tipos de contribuições, tais como serviços, informações, conhecimento e coleta de dados provenientes de um grupo de pessoas. O uso de *crowdsourcing* para processamento multimídia não é exclusiva deste trabalho. De fato, seu uso já é conhecido pela comunidade internacional de multimídia[1] e nesta tese, busca-se uma investigação mais aprofundada da aplicação do conceito *crowdsourcing* para o processamento de vídeos.

[1] *Crowdsourcing in Multimedia*: http://www.crowdmm.org/

Técnicas de *crowdsourcing* podem ser utilizadas de diferentes formas no domínio da multimídia. Alguns exemplos são (i) a criação de datasets de imagens, de vídeos e áudios; (ii) a avaliação da qualidade de codificadores de vídeo e (iii) a avaliação de resultados produzidos por técnicas de processamento automático de conteúdos. Outros trabalhos, apresentados no capítulo 3, utilizam as técnicas para resolver problemas específicos como anotação de imagens, sumarização de vídeos e outras abordagens.

Os exemplos anteriores ilustram a flexibilidade de aplicação de *crowdsourcing* para solucionar problemas da área de multimídia. No caso desta tese, *crowdsourcing* é aplicado para resolver o problema de sincronização de vídeos disponibilizados livremente na Internet por usuários e produzidos por eles de forma voluntária e não coordenada durante um evento social limitado no tempo e no espaço.

1.1.OBJETIVO

O foco principal desta tese é aplicar uma abordagem *crowdsourcing* para sincronizar um conjunto de vídeos gerados por usuários durante um mesmo evento social. Nesta tese, considera-se que o termo evento social designa um acontecimento compartilhado grupo de pessoas e que ocorre em um determinado lugar, durante um determinado intervalo de tempo.

O escopo desta tese se restringe aos vídeos capturados por usuários de forma voluntária, não necessariamente coordenada, numa certa localidade e durante certo intervalo de tempo. A sincronização destes vídeos permite não só recontar a história do evento, mas também oferecer diferentes opções de visualização deste evento a partir das diversas capturas provenientes dos usuários.

Não fazem parte do escopo do trabalho as questões relacionadas à recuperação e ao agrupamento dos vídeos associados a um mesmo evento. Em outras palavras, considera-se que o conjunto de vídeos obtidos em um mesmo evento social, que funciona como a entrada para o processo proposto nesta tese, foi obtido numa fase anterior à sincronização.

O resultado do processo de sincronização de vídeos gerados pelos usuários, interesse principal deste trabalho, gera a especificação dos instantes de tempo a

partir dos quais estes vídeos passam a retratar um conteúdo similar, correlacionado no tempo e no espaço. Sendo assim, também está fora do escopo desta tese o estudo aprofundado das possíveis formas de apresentação dos vídeos após a conclusão do processo de sincronização.

Dentro do escopo discutido, o objetivo principal desta tese **é propor e avaliar um modelo de processamento *crowdsourcing*, demonstrando que ele pode ser usado com sucesso para a sincronização de vídeos correlacionados a um mesmo evento social e gerados voluntariamente por usuários**.

1.2. QUESTÕES DE PESQUISA

Após a definição clara do objetivo desta tese, é importante destacar as etapas seguidas durante o seu desenvolvimento. Cada uma das etapas foi associada a um dos desafios encontrados, traduzidos sob a forma de importantes questões de pesquisa.

1. *Como sincronizar UGVs, considerando um cenário distribuído e falta de informações de sincronização explícita destes vídeos?*

Existem diversas formas de sincronizar, especificar, e depois apresentar estes vídeos de maneira sincronizada. No cenário de UGVs, os vídeos são armazenados e disponibilizados em diferentes servidores, independentes, e sem nenhuma identificação sobre marcações temporais que auxiliem a sincronização. Assim, é proposta uma solução que permita sincronizar múltiplos vídeos para diversos usuários, onde cada usuário assiste a estes vídeos de forma independente, em seu próprio dispositivo, sendo estes vídeos transmitidos em diferentes canais, com atrasos e capacidades variados e sem nenhum tipo de marcação temporal associada aos conteúdos destes vídeos.

2. *É possível sincronizar UGVs com o uso de técnicas de crowdsourcing?*

Muitos trabalhos na literatura mostram como utilizar a *crowd* na anotação, sumarização, criação e outras tarefas envolvendo vídeos, porém não existem trabalhos que utilizem a *crowd* no processo de sincronização. Desta forma é preciso descobrir se com o uso das técnicas de *crowdsourcing* é possível que a *crowd* gere

contribuições que, ao serem agregadas, criem uma especificação de sincronização coerente para o conjunto de vídeos processados.

3. *Como é definido este método de sincronização de vídeos utilizando a crowd?*

A utilização de técnicas de *crowdsourcing* não é simples como a importação de uma biblioteca ou a execução de um programa. A utilização da *crowd* exige que diversos aspectos únicos sejam considerados, a saber: como conseguir os membros para constituir uma *crowd*? Como saber se as contribuições destes membros são válidas? Estes são apenas alguns dos desafios da inclusão da computação humana no processo, sendo necessário descrever como estes desafios podem ser resolvidos.

4. *O uso de uma abordagem híbrida pode auxiliar o problema da sincronização dos vídeos pela crowd?*

Ambas as abordagens puramente humana e automática podem apresentar limitações, desta forma a utilização das duas abordagens em um método híbrido pode fazer com que uma sobreponha às limitações da outra, melhorando assim os resultados que ambas, de forma isolada, poderiam atingir. Para isso deve ser testada como uma abordagem que se utilize de etapas baseada em processamento humano e de etapas automáticas se comporta.

1.3.ESTRUTURA DA TESE

Diante da problemática geral da pesquisa, este trabalho foi estruturado da seguinte forma:

O segundo capítulo apresentará os resultados da pesquisa sobre o processo de sincronização a ser utilizado no trabalho. Nele é descrito o processo de sincronização, que consiste na formalização das relações temporais entre os objetos de mídia no cenário desta tese. Com este capítulo é esperado que se responda à primeira pergunta da tese: *Como sincronizar UGVs, considerando um cenário distribuído e falta de informações de sincronização explícita destes vídeos?*

No capítulo 3, os conceitos fundamentais de computação humana e *crowdsourcing* serão discutidos com maior profundidade. Em seguida, será descrito o resultado de um mapeamento sistemático sobre trabalhos que utilizaram técnicas

de *crowdsourcing* para processamento de vídeos nas mais diversas áreas, com os principais resultados e observações.

O capítulo 4 aborda diretamente o uso da *crowd* na tarefa de sincronização de vídeos. Ele é o resultado do conhecimento adquirido pela pesquisa apresentada no capítulo 3 e dos experimentos desenvolvidos para aplicação do uso da *crowd* como um método de sincronização. No capítulo são descritas as etapas que o método possui, os desafios de gerenciamento relacionados ao uso da *crowd*, e os tipos de tarefas que a *crowd* pode realizar. Ao fim deste capítulo, espera-se responder à questão: *É possível sincronizar UGVs com o uso de técnicas de crowdsourcing?*

O capítulo 5 irá mostrar os principais pontos que devem ser levados em consideração no processo de sincronização de vídeos. Ao contrário do capítulo 3 e 4, que focam em mostrar que o processo proposto é viável, o capítulo 5 discute a construção de ferramentas para diversas aplicações para o processamento de vídeos pela *crowd* através da apresentação de um modelo funcional.

O capítulo 6 apresenta um experimento de simulação de um sistema de *crowdsourcing* para a sincronização de vídeos, demonstrando diversos aspectos que podem ter impacto no uso do método.

Ao final dos capítulos 5 e 6, espera-se que o leitor tenha a resposta à questão: *Como é definido este método de sincronização de vídeos utilizando a crowd?*

No capítulo 7, serão discutidos os principais experimentos realizados ao longo da pesquisa. Um dos experimentos se reporta à ideia de processamento híbrido, mesclando a abordagem automática, puramente computacional com computação humana, baseada em *crowdsourcing*, dos vídeos a serem sincronizados. O capítulo também descreve os estudos de caso usando quatro *datasets* distintos para avaliar o método híbrido proposto. Ao concluir o capítulo, espera-se que a seguinte questão seja respondida: *O uso de uma abordagem híbrida pode auxiliar o problema da sincronização dos vídeos pela crowd?*

Além dos experimentos focados na verificação do método híbrido, o capítulo 7 descreve e analisa dois experimentos de métodos baseados na utilização pura da *crowd* para a sincronização de vídeos, utilizando tanto uma *crowd* contratada, quanto uma *crowd* especializada voluntária.

Concluindo o texto, o capítulo 8 apresenta considerações finais sobre a pesquisa, além dos resultados acadêmicos alcançados e trabalhos futuros que devem ser desenvolvidos para a continuidade desta pesquisa.

2. SINCRONIZAÇÃO DE VÍDEOS GERADOS POR USUÁRIOS

Nesta tese, assume-se que sincronizar é a ação de coordenar a apresentação de dois ou mais objetos em um dispositivo de apresentação comum e compartilhado. Em multimídia, se dois objetos de mídia estiverem sincronizados, os observadores dessa apresentação composta perceberão a apresentação como um objeto único, cujo conteúdo evolui de forma síncrona no tempo e no espaço.

2.1. SINCRONIZAÇÃO MULTIMÍDIA

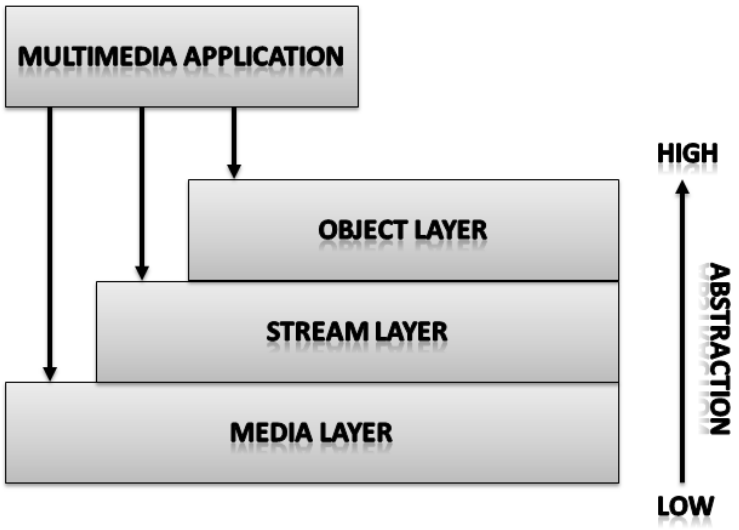
Os sistemas multimídia permitem a integração de fluxos de dados de diferentes tipos, incluindo mídias contínuas (áudio e vídeo) e discretas (texto, dados, imagens). A sincronização é essencial para a integração dessas mídias em uma apresentação multimídia. Na sincronização multimídia, é comum o uso da taxonomia criada por Meyer et al. (1993) para classificação da sincronia.

Esta classificação é baseada em camadas de abstração multimídia (Figura 1), onde na camada de mídia (*Media Layer*) uma aplicação opera em um único fluxo contínuo de mídia, que é tratado como uma sequência de *LDUs/MDUs* (unidades de dados lógicos/unidades de dados de mídia); a camada de fluxo (*Stream Layer*) permite que a aplicação opere em fluxos de mídia contínuos, bem como em grupos de fluxos; a camada de objeto (*Object Layer*) permite a especificação mais exata de sequências de apresentação, onde cada objeto de mídia se relaciona com um eixo de tempo e define uma sequência de eventos.

Na camada de mídia, a sincronização dentro do fluxo (*intrastream*) lida com a manutenção, durante a reprodução, da relação temporal dentro de cada fluxo de mídia dependente do tempo, ou seja, entre o *MDUs* do mesmo fluxo. Na camada de fluxo, a sincronização entre fluxos (*inter-stream*) refere-se à sincronização, durante a reprodução, dos processos de reprodução de diferentes fluxos de mídia envolvidos na aplicação e a sincronização ao vivo, lida com a apresentação de informações para que sejam apresentadas da mesma forma que foram coletadas.

A camada de objeto apresenta sincronização sintética onde várias partes de informações (objetos de mídia), no tempo de apresentação, devem ser devidamente ordenadas e sincronizadas no espaço e tempo.

Figura 1 - Camadas de Abstração de Sincronização



Fonte: reprodução de Meyer et al. (1993)

Um mosaico composto por diversos vídeos capturados em paralelo por múltiplas câmeras ilustra um exemplo de aplicação do conceito de sincronização. Estes vídeos mostram a mesma cena, mas de diferentes pontos de vista até mesmo opostos. Se eles são sincronizados, o espectador pode, a qualquer momento, escolher assistir as cenas capturadas do ponto de vista de uma ou de múltiplas câmeras, sem perder a sensação de continuidade do conteúdo.

No entanto, assim como as apresentações e cenários multimídia evoluíram, os modelos também tiveram que se adaptar. Por exemplo, modelo de camadas de Meyer et al. (1993) não contempla o cenário de sincronização entre múltiplos destinos, onde diversas apresentações devem estar síncronas entre si, mas em diversos dispositivos diferentes. Huang et. al. (2013) expande o modelo de Meyer levando em considerações estes estas mudanças: o modelo multidimensional (Figura 2). Este modelo é dividido em três dimensões:

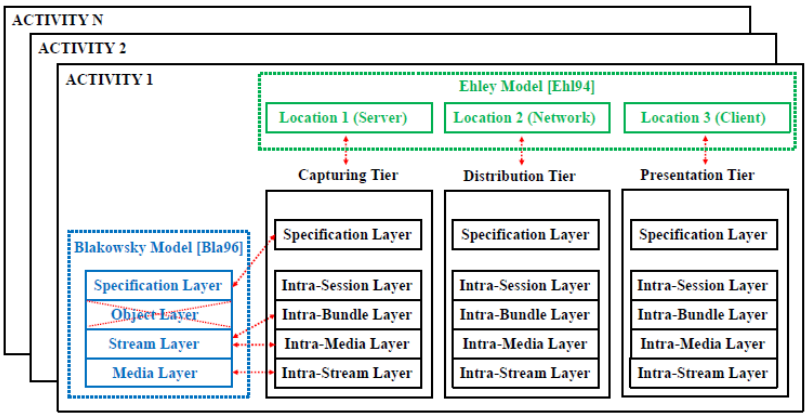
1) A dimensão da heterogeneidade de escala e dispositivo. A camada de objeto do modelo tradicional é removida, pois eles só se concentram em fluxos multimídia contínuos. A camada de especificação permanece para a formulação da especificação das várias camadas;

2) A dimensão dos controles de sincronização em múltiplos locais. O modelo de sincronização multidimensional adiciona controles de sincronização em cada local, juntamente com suporte temporal para um grande número de dispositivos heterogêneos;

3) A dimensão da sincronização dependente da aplicação. Esta dimensão descreve o impacto da heterogeneidade da aplicação na percepção da sincronização humana.

Não é possível usar referências de sincronização uniforme para representar uma plataforma multimídia. Cada aplicativo desenvolvido para uma plataforma deve identificar suas próprias referências baseadas na funcionalidade de atividades executadas e requisitos do usuário final.

Figura 2 – Modelo de Sincronização Multidimensional



Fonte: reprodução de Huang et al. (2013)

A proposta de sincronização apresentada neste trabalho pressupõe o correto funcionamento das camadas de *Media* e *Stream* nos players utilizados, trabalhando diretamente com a sincronização sintética dos objetos de vídeo na dimensão de sincronização. Esse tipo de sincronização também incorpora aspectos semânticos da sincronização, onde a percepção da sincronização é afetada pelo significado dos conteúdos e interpretação do usuário. Esse nível de sincronização pode ser interpretado como uma camada acima dos modelos apresentados: uma camada semântica (SEGUNDO e SANTOS, 2014). A camada semântica permite a comunicação, busca, recuperação e interpretação de *playouts* e seus conteúdos, lidando com além da sincronização entre destinos, com a sincronização contextual (conteúdos *CrossMedia*, *mash-ups*, etc).

2.2.CENÁRIO DE SINCRONIZAÇÃO

Numa visão simplificada, o contexto desta tese consiste em um sistema no qual um **conjunto de dados de entrada** deve ser **processado** para produzir uma **saída** esperada. Neste caso:

- Os dados de entrada são os vários vídeos, que contém cenas sobrepostas de um evento (algo que ocorre em um determinado lugar, durante um determinado intervalo de tempo);
- O processamento consiste em encontrar os pontos de sincronização (se existirem) entre os vídeos da entrada;
- A saída esperada consiste em uma especificação de todas as relações temporais entre os vídeos do conjunto de entrada; esta especificação pode ser usada para alimentar outras aplicações que implementam diferentes formas de visualização dos conteúdos dos vídeos processados.

Os passos descritos definem um processo geral de sincronização de vídeos, porém, nesta tese, considera-se que (i) o conjunto de entrada será restringido aos vídeos gerados por usuários a partir de um evento social e (ii) o uso de computação humana na fase de processamento dos vídeos.

Os UGVs são um tipo de conteúdo multimídia criado e compartilhado na Internet por diferentes tipos de usuários, de forma voluntária, utilizando dispositivos heterogêneos, sem o uso de qualquer mecanismo de coordenação explícita, seja na sua captura, seja na distribuição destes conteúdos.

Estes *UGVs* podem estar correlacionados em torno de eventos tais como protestos, festivais de música ou atividades esportivas, resultando em conteúdos melhorados que podem ser reutilizados de diferentes maneiras. Além disso, cada um desses vídeos revela um ponto de vista único sobre o que está acontecendo, de acordo com a identidade do usuário e crenças (em termos de ideologia, equipe e identificação de grupo, etc), bem como o contexto do usuário e preferências (em termos de posicionamento, capacidades de dispositivo e limitações, etc.)

Neste tipo de cenário, é impossível garantir que todos os *UGVs* relacionados a um mesmo momento significativo de um evento social sejam homogêneos em termos de qualidades visual e aural, ou mesmo, que este momento tenha sido

capturado por um determinado usuário. Desta forma, com cenários tão heterogêneos, não há, no estágio atual da computação, um processo de sincronização para estes vídeos que seja puramente computacional, suficientemente genérico e independente das características do conjunto de vídeos de entrada. O trabalho de (DOUZE, REVAUD, *et al.*, 2016) ilustra esta situação, demonstrando como as características do conjunto de entrada, tais como ângulo de captura e qualidade dos vídeos, impactam diretamente na porcentagem de pontos de sincronização encontrados a partir do processamento dos vídeos.

O problema de sincronizar *UGVs* correlacionados a um determinado tópico ou evento pode ser enunciado como sendo um problema de narrativa (*storytelling*), no qual se quer recontar a história de um evento a partir das diversas filmagens realizadas durante o seu desenrolar. Para isto, é preciso alinhar (sincronizar) os vídeos, de forma que quando apresentados, eles complementem um ao outro, exibindo diferentes pontos de vista e detalhes que foram capturados por um, mas não necessariamente pelos outros vídeos do conjunto de entrada. A partir da sincronização é possível ainda permitir ao espectador escolher qual, dentre as diversas câmeras sincronizadas, ele deseja assistir a cada momento de apresentação.

Resolver o problema de sincronização de conteúdos multimídia consiste em: (i) inicialmente, posicionar todos os conteúdos correlacionados em uma referência temporal global e, em seguida, (ii) apresentar estes conteúdos de forma síncrona, a partir da referência temporal global, a fim de produzir uma narrativa coerente.

Existem vários métodos, como será mostrado na seção 2.4.4, que podem ser utilizados para que o conjunto de vídeos de entrada tenha uma referência temporal única e global, facilitando o processo de reprodução sincronizada destes conteúdos, como: o uso de dispositivos que, durante a captura, criam pontos de sincronização entre vídeos (câmeras sincronizadas); softwares que recebem um conjunto de vídeos como entrada e, em seguida, geram saídas sincronizadas (soluções automáticas); e softwares para edição que são usados por profissionais para definir os links entre os vídeos.

Por outro lado, como *UGVs* são normalmente disponibilizados em repositórios de vídeos Web e sua produção é feita de forma descoordenada, não se pode garantir a existência de uma referência temporal única entre eles. Porém, mesmo

que fosse possível garantir que todos os vídeos armazenados possuíssem uma garantia de sincronização global na origem (por exemplo, se os relógios de todos os *smartphones* usados na captura de vídeos em um evento estivessem completamente sincronizados sob um relógio único e global), ainda assim, como estes vídeos podem estar armazenados em diferentes repositórios, a latência e o *jitter* na transmissão de cada vídeo poderia variar de usuário para usuário, levando a uma apresentação sem sincronismo no destino.

A próxima seção apresenta um método de sincronização de múltiplos conteúdos, provenientes de múltiplas fontes, num cenário equivalente ao do escopo desta tese, no qual estes conteúdos são *UGVs*.

2.3.INTEGRAÇÃO DE MÚLTIPLOS CONTEÚDOS

Uma apresentação de vídeo tradicional envolve um dispositivo de usuário (*UD – User Device*) que é capaz de decodificar e apresentar um único conteúdo (conteúdo principal ou *MC – Main Content*) originado de uma fonte única (provedor do *MC*). Em um cenário não-tradicional (HUANG, NAHRSTEDT e STEINMETZ, 2013), o ambiente de apresentação pode ser composto de vários dispositivos de usuário (TV, computador, *smartphones*, *tablets*, etc) capazes de apresentar vários conteúdos entregues a partir de várias fontes.

Nessa situação, o usuário acessa uma combinação (*mashup*) de conteúdos digitais que precisa de uma especificação da sincronização definida para orquestrar a apresentação combinada. *Mashups* são aplicativos gerados pela combinação de conteúdo, apresentação ou outras funcionalidades de aplicativos de fontes díspares. Eles têm como objetivo combinar essas fontes para criar novos aplicativos ou serviços úteis aos usuários (YU ET AL., 2008).

Seguindo o exemplo de um cenário de transmissão de vídeo ao vivo, um espectador pode adicionar outros serviços (outros fluxos de vídeo) e conteúdos extras que possam melhorar sua experiência enquanto assiste a transmissão. Exemplos típicos de serviços são legendas e fluxos de áudio de diferentes idiomas relacionados ao conteúdo principal. Diversas fontes podem ser utilizadas para prover o conteúdo principal, como: *CDNs*, *P2P*, vídeos locais, servidores de *streaming*, etc. Nesta situação, o espectador deve ser capaz de combinar diferentes serviços e criar um *mashup* de seu conteúdo: assistir a um filme fornecido pelo seu serviço de

streaming; usando legendas nativas ou importando a partir de um servidor Web com novas linguagens, ou serviços inclusivos, como no caso de áudio-descrição (ENCELLE, OLLAGNIER-BELDAME, *et al.*, 2011) ou do uso de linguagem de sinais (ARAÚJO, FERREIRA, *et al.*, 2014).

Esta combinação de serviços e conteúdos, no entanto, traz o seguinte problema: *Como sincronizar múltiplos conteúdos para cada usuário em seu ambiente, uma vez que o conteúdo é transmitido através de diferentes canais e de diferentes fontes que não possuem uma especificação de sincronização explícita*? Para isto é necessário um processo de sincronização para encontrar os pontos de sincronização entre os conteúdos, e especificar e disponibilizar a especificação da sincronização.

A seção seguinte detalha com é possível criar uma especificação da sincronização dos conteúdos dos vídeos, considerando que existe um método genérico capaz de encontrar os pontos de sincronização entre eles.

2.4.ALINHAMENTO DE *TIMELINES*

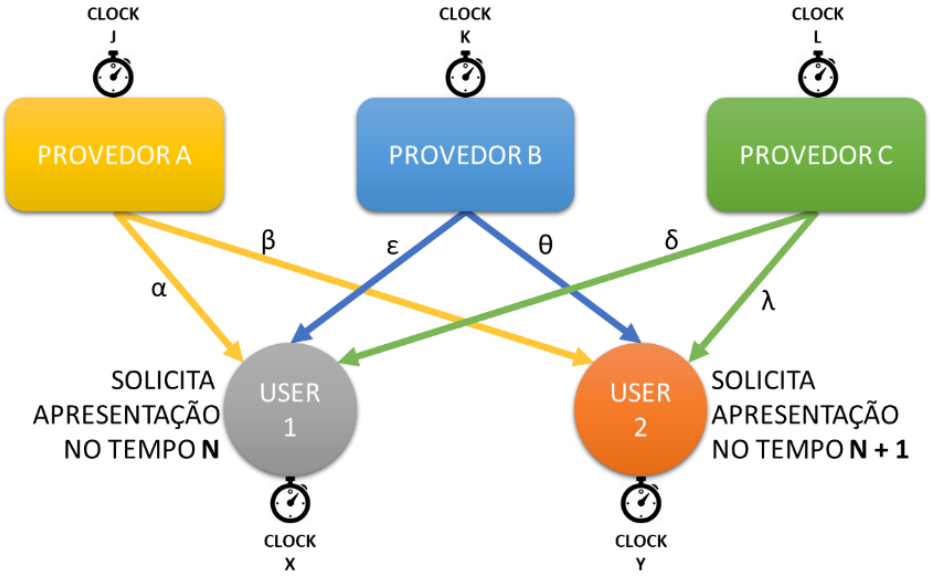
O problema endereçado pode ser entendido como uma questão de sincronização de relógios para a apresentação síncrona de objetos multimídia: pretende-se fornecer uma solução para sincronizar o conteúdo de várias fontes em cada ambiente de apresentação para cada usuário. A solução do problema em um sistema centralizado seria trivial. Bastaria utilizar um servidor centralizado, que iria impor um relógio global para todos os componentes do sistema. No cenário proposto o problema é mais complexo, pois a princípio, não há nenhum componente central que funciona como uma marca de tempo para todos os dispositivos e conteúdos produzidos. Assim, cada usuário é independente dos demais, sincronizando, conteúdos em seu próprio ambiente, isto é, de acordo com seu próprio relógio local.

Vale notar que um relógio global não consegue resolver sozinho o problema da sincronização no local de apresentação. Mesmo em uma situação na qual um mesmo vídeo é acessado por diferentes usuários, cada uma das cópias desses conteúdos será entregue a estes usuários em instantes diferentes devido às variações nos atrasos de transmissão (*buffering, jitter*, ...). Na solução proposta, o problema é contornado a partir do alinhamento da *timeline* de todos os conteúdos a serem apresentados no dispositivo do usuário, ou seja, no local (destino) de

apresentação. Isso exige que apenas a relação entre os vídeos seja conhecida previamente, sem necessidade de conhecimento dos valores de atraso de propagação desde a origem até o destino de apresentação para cada fluxo de vídeo a ser sincronizado, já que estes valores são dinâmicos para cada usuário que solicita os vídeos. Um relógio local leva em conta os atrasos armazenados para garantir a correta apresentação do *mashup* de conteúdos com relação ao tempo.

O desafio é fazer com que os fluxos de vídeo provenientes de diferentes fontes (e com diferentes referências temporais) estejam síncronos para cada usuário. No esquema da Figura 3, note que usuários e provedores possuem relógios próprios (X, Y, J, K, L), ou seja, não há um relógio global. A figura também mostra que os diferentes atrasos de transmissão (representados pelas letras gregas α, β, etc.) são variáveis de cada provedor para cada usuário, de forma que é impossível saber o tempo de atraso, já que não há um relógio em comum entre os clientes e servidores. O atraso é o tempo que o vídeo leva desde a requisição de solicitação no provedor até o início da apresentação no player do cliente, abrangendo atrasos de transmissão, *buffering* e outros. O método deve então permitir que mesmo assim os conteúdos sejam apresentados de forma sincronizada no usuário. Isso será possível por que todas as informações de sincronização necessárias estarão presentes em cada usuário, de forma que ela irá independer dos demais valores (SEGUNDO e SANTOS, 2015). Mesmo que ocorram atrasos após a transmissão já ter tido início, ainda assim é possível re-sincronizar, pois o cliente possui controle do player e é capaz de identificar a pausa na transmissão, sendo possível logo em seguida executar novamente a sincronização.

Figura 3 – Múltiplos provedores de conteúdo, transmitindo seus vídeos para usuários. Cada fluxo de vídeo possui um atraso diferente até chegar ao cliente, e cada entidade possui um relógio próprio



Fonte: Elaborada pelo autor

O método de sincronização proposto a seguir permite que qualquer provedor de conteúdo, de forma independente dos demais, possa ter seu vídeo sincronizado com os demais vídeos de um determinado evento, sem a necessidade dele ou dos outros provedores proverem informações explícitas de sincronização relacionadas aos seus vídeos.

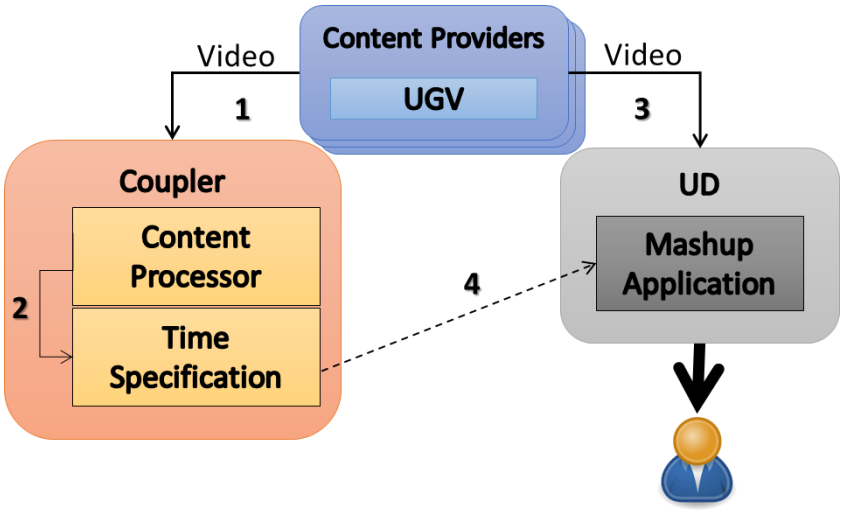
Esse método assume que:

- Qualquer provedor de conteúdo pode fornecer conteúdo sincronizado para o *mashup* dos clientes, sem depender de uma especificação de tempo explícita enviada por qualquer outro provedor de conteúdo;
- O método pode ser implementado no nível de aplicação;
- Vários vídeos podem ser sincronizados localmente em um mesmo ambiente destino, sem comunicação direta entre os provedores destes conteúdos;
- Qualquer cliente pode ter acesso aos vídeos disponibilizados e sincronizados;
- Não é possível garantir que os relógios locais de cada cliente estejam em sincronia, isto é, nenhuma entidade é previamente sincronizada com outra.

É importante enfatizar que a abordagem proposta não visa sincronizar todos os fluxos em um relógio global e, portanto, não existindo preocupação com a solução de sincronização entre os vários destinos (*IDMS*) (MONTAGUD e BORONAT, 2012). A sincronização dos diversos fluxos de vídeo é garantida em relação a um relógio local, que dita o ritmo da apresentação do *mashup* que combina esses conteúdos localmente em cada usuário de forma independente.

A Figura 4 apresenta as três entidades principais que compõem o modelo proposto para sincronização de conteúdos de várias fontes. Os *content providers* transmitem fluxos de vídeo para as demais entidades (1). O acoplador recebe os vídeos, os processa e gera a especificação de tempo que é armazenada (2). Essa especificação é então requisitada pelos *UDs* após solicitar os vídeos (3). A transmissão (4) da especificação pode ser realizada através de uma requisição HTTP que transmite do acoplador para o *UD* um arquivo *JSON* contendo a especificação.

Figura 4 - Entidades envolvidas no método de Sincronização



Fonte: Elaborada pelo autor

2.4.1. Provedor de Conteúdo

O provedor de conteúdo (*Content Provider – CP*) armazena os *UGVs*. A aplicação *mashup* e o acoplador acessam um vídeo a partir da sua *URI* para apresentá-lo a um espectador ou para processar seu conteúdo. Note que o provedor de conteúdo não provê nenhuma informação de sincronização junto com os vídeos.

2.4.2. Dispositivo do Usuário

O Dispositivo do Usuário (*User Device – UD*) é o componente responsável pela apresentação sincronizada dos vídeos nos clientes. Ele executa a aplicação denominada *mashup*, que é responsável por consumir os serviços fornecidos pelo *coupler* e *CPs*. O *mashup* também recebe e processa os vídeos (enviados pelos *CPs*) e as informações de *offset* (enviadas pelo *coupler*), gerando composições sincronizadas destes vídeos.

Para reproduzir dois vídeos sincronamente, o *mashup* inicializa a apresentação de cada vídeo e calcula a diferença atual entre os seus conteúdos, uma vez que a inicialização de cada vídeo pode não ocorrer exatamente no mesmo instante. Em seguida, a partir o *offset* fornecido pelo *coupler,* ele atrasa a apresentação do vídeo adiantado tempo suficiente para que o *offset* seja atingido. A partir daí os conteúdos estarão sincronizados. Caso seja verificada uma perda de sincronização após a apresentação combinada ter sido iniciada, será necessário recalcular o *offset* e realizar uma nova sincronização dos vídeos.

2.4.3. Acoplador

O acoplador (do inglês *Coupler*) é responsável por fornecer as especificações temporais entre os múltiplos vídeos providos pelos *CPs* a um *mashup* que combina estes conteúdos. O acoplador processa os vídeos recebidos a fim de encontrar os pontos de sincronização entre eles.

Cada ponto de sincronização resultante do processamento do conjunto de vídeos é chamado de acoplador. Cada acoplador armazena a diferença de tempo (*offset*) entre a ocorrência de uma mesma situação singular capturada por um par de vídeos correlacionados (por exemplo, uma mesma cena capturada em ângulos diferentes ou o som de uma mesma explosão capturada por duas câmeras diferentes). Esta diferença é determinada em função do relógio local do *coupler*. Para realizar a sincronização dos *mashups* combinando os vídeos nos locais de apresentação, os clientes utilizam os valores de *offset* armazenados pelos *couplers* e enviados aos clientes.

São funcionalidades do acoplador:

Armazenamento e recuperação dos *offsets*: permitir a adição, atualização e recuperação de qualquer *offset* (Δ) para qualquer par de objetos em tempo *O*(*N*);

Inferência de relações temporais: possuir métodos de inferência que usam valores armazenados para descobrir valores de relações temporais ainda desconhecidas. O uso da transitividade pode ser um destes métodos. Por exemplo, se as relações *AB* e *AC* são conhecidas, é possível inferir *BC*, conforme a Equação (4).

$$\Delta_{B,C} = begin(\mathrm{B}) - begin(\mathrm{C}) \quad (1)$$

$$\Delta_{A,B} = begin(\mathrm{A}) - begin(\mathrm{B}) \rightarrow begin(\mathrm{B}) = begin(\mathrm{A}) - \Delta_{A,B} \quad (2)$$

$$\Delta_{A,C} = begin(\mathrm{A}) - begin(\mathrm{C}) \rightarrow begin(\mathrm{C}) = begin(\mathrm{A}) - \Delta_{A,C} \quad (3)$$

Substituindo (2) e (3) em (1):

$$\Delta_{B,C} = begin(\mathrm{A}) - \Delta_{A,B} - (begin(\mathrm{A}) - \Delta_{A,C})$$

$$\Delta_{B,C} = begin(\mathrm{A}) - \Delta_{A,B} - begin(\mathrm{A}) + \Delta_{A,C}$$

$$\Delta_{B,C} = \Delta_{A,C} - \Delta_{A,B} \quad (4)$$

Geração de apresentações sincronizadas: é possível criar uma apresentação para o evento com o uso dos valores de Δ armazenados. Diferentes parâmetros podem ser utilizados para a geração destas apresentações, como por exemplo, gerar uma apresentação com a maior duração possível ou com a melhor qualidade possível, a partir do conjunto de vídeos de entrada.

Processamento de Conteúdo:

Diversas técnicas podem ser aplicadas para identificar pontos de sincronização entre vídeos. Algumas utilizam informações adicionadas ao conteúdo dos vídeos durante a captura, tais como marcas d'água nas trilhas de áudio e/ou nos *frames* de vídeo e *timestamps* adicionadas sob a forma de metadados. Outras se baseiam em mecanismos como áudio (GUIMARÃES, CESAR, *et al.*, 2011) e/ou vídeo *fingerprinting* (HOWSON, GAUTIER, *et al.*, 2011), e processamento dos conteúdos dos vídeos (FINK, COVELL e BALUJA, 2006), além de métodos alternativos como o uso de *crowdsourcing*. A seção a seguir apresenta mais detalhes de técnicas que

podem ser utilizadas para o processamento dos vídeos, resultando na identificação de pontos de sincronização.

2.4.4. Processamento de Conteúdo

Existem diferentes formas de gerar a sincronização entre dois vídeos, como por exemplo fazendo a análise de áudio e frame. Porém, existem alternativas a serem exploradas para tal, como o uso de técnicas de computação humana e ainda a combinação de ambas abordagens dando origem a uma técnica híbrida.

2.4.4.1. Métodos de processamento de *frames*

Wang *et al.* (2014) sincronizam vídeos no espaço e tempo, permitindo a navegação entre eles através da busca de semelhanças entre os vídeos e suas *timelines*. A sincronização funciona para vídeos oriundos de câmeras diferentes. Eles conseguem a sincronização obtendo uma aproximação da qualidade de alinhamento entre pares de clipes, calculado como um histograma ponderado de características.

Floria *et al.* (2013) apresentam tanto um novo método de sincronização, como também uma seção detalhada de estado da arte com vários trabalhos relacionados a sincronização de vídeos. O método apresentado analisa as diferenças entre frames usando uma versão aprimorada do algoritmo *ConCor* para encontrar pontos de sincronização entre os vídeos.

Wang (2016) apresenta um método para sincronizar duas câmeras no tempo, sendo que elas se movem de forma independente, com visões sobrepostas. Ele usa variações temporais entre *frames* (como objetos em movimento) para o alinhamento dos vídeos. Ele também gera imagens de pulso rastreando objetos em movimento e examinando as trajetórias para mudanças de velocidade. Em seguida, analisa a qualidade do alinhamento para todos os pares de frames e identifica os pontos de sincronização.

Douze *et al.* (2016) aborda o problema da recuperação específica de eventos dentro do vídeo. Dado um vídeo como forma de consulta, o objetivo é recuperar outros vídeos do mesmo evento que se sobrepõem temporalmente. Para isso, eles precisam gerar a especificação de sincronização como saída. Na abordagem, os

autores codificam os descritores dos frames de um vídeo para representar sua aparência e ordem temporal. Seu algoritmo alinha os vídeos em uma linha de tempo global que permite a reprodução síncrona dos vídeos de uma determinada cena.

Um fato comum entre todos os métodos automáticos citados é que seus desempenhos são fortemente dependentes de como o conteúdo foi capturado e codificado. De acordo com Bano e Cavallaro (2015), os métodos automáticos de sincronização baseados em *frames* de vídeo têm alguns importantes desafios a serem considerados:

- *Wide baselines*: as abordagens que dependem de características de textura combinadas sofrem com a invariância limitada pelas mudanças da visão das câmeras. Por isso, as abordagens baseadas em características não sofrem apenas cenas em que as câmeras ficam opostas uma à outra, mas também cenas com visões superpostas em grandes ângulos;
- Movimento da câmera: uma câmera "trêmula" pode introduzir erros diversos no processo de sincronização. No entanto, a aplicação da sincronização de vídeos em eventos de perspectiva múltipla capturados por usuários requer um algoritmo que possa lidar também com esse tipo de cenário;
- Dinamismo dos planos de fundo: um plano de fundo em mudança pode confundir qualquer algoritmo de sincronização na identificação do objeto de real interesse;
- Oclusões: especialmente nas abordagens que envolvam o rastreamento de objetos e/ou de suas características, o desaparecimento temporário de objetos de interesse, bem como oclusões de múltiplos objetos em movimento pode ser desastroso para a sincronização, caso ela esteja baseada nestes objetos.

2.4.4.2. Métodos de processamento de trilhas de áudio

Su *et al.* (2012) apresentam a sincronização de vídeos usando a análise de áudio. *Fingerprintings* são geradas para cada áudio, e uma comparação entre os

vídeos é realizada, considerando o número de *fingerprintings* semelhantes: quando há mais ocorrências, as chances de sincronização também aumentam.

Bano e Cavallaro (2015) mostram outro método para sincronizar vídeos com base no áudio. Sua abordagem usa a análise *Chroma* do áudio, agrupando e sincronizando áudios de um mesmo evento. Eles usam um tipo específico de vídeo como entrada: os vídeos gerados por usuários.

Guggenberger (2015) apresenta o *Aurio*, uma biblioteca de software de código aberto escrita para processamento, análise e recuperação de áudios (*github.com/protyposis/Aurio*). O uso do *Aurio* permite analisar o áudio para diversas propostas, como a sincronização de vídeo usando o processamento de áudio. A interface *AudioAlign* usa a biblioteca *Aurio* para sincronizar automaticamente as gravações de áudio e vídeo, utilizando algoritmos de *fingerprinting* de áudio.

De maneira similar aos métodos automáticos baseados nas características visuais dos vídeos, os métodos que utilizam o processamento das trilhas de áudio para suporte à sincronização entre vídeos também têm desafios a serem vencidos. O primeiro está relacionado à ausência de áudio ou à baixa intensidade do sinal sonoro devido a problemas na captura. Isso ocorre quando os microfones da câmera estão longe do foco de interesse da filmagem ou quando eles não conseguem capturar o áudio com intensidade suficientemente alta para permitir o seu correto processamento. Outros problemas estão relacionados ao ruído do sinal de áudio capturado, à captura de áudios extras e à não captura do áudio dependendo da localização da câmera durante a filmagem de um evento.

2.4.4.1. Método de Computação Humana

Há atividades que são triviais para a maioria das pessoas, mas que são complexas quando tratadas computacionalmente. Estes tipos de tarefas têm características que requerem criatividade, abstração e adaptação que, até o momento, são características comuns aos humanos, mas complexas às máquinas. Descrição de imagens, especialmente quando há oclusão ou envolve análise subjetiva; autoria de conteúdo; análise de emoções, entre outras, são exemplos de problemas que requerem a inteligência humana para a sua solução.

De forma análoga, é possível fazer uso de métodos de computação humana para a sincronização de vídeos. O método de sincronização humana utiliza as pessoas para identificar os pontos de sincronização entre pares de vídeos.

Este método é uma das contribuições desta tese e começa a ser apresentado do Capítulo 2, onde é apresentado o conceito de computação e como utiliza-la, sendo seguido nos outros capítulos pela apresentação da proposta de sincronização pelas pessoas.

2.4.4.2. Métodos Híbridos

As abordagens híbridas combinam métodos de processamento automático realizados pela máquina, com técnicas de computação humana, como o *crowdsourcing*, para solucionar um problema. Alguns trabalhos recentes têm integrado contribuições da *crowd* na tentativa de contornar problemas enfrentados pelos métodos automáticos e também, de melhorar os seus desempenhos.

Albarqouni (2016) apresenta a combinação de contribuições de *crowdsourcing* e algoritmos de aprendizagem de máquina para auxiliar a detecção de câncer de mama em imagens de histologia. As contribuições da *crowd* são usadas como *feedback* para os algoritmos de aprendizado de máquina. Do mesmo modo, Fan (2014) usa a *crowd* para alimentar um algoritmo de aprendizado de máquina, no entanto, essa abordagem é usada para combinar tabelas de informações disponíveis na Web.

As abordagens híbridas não se limitam ao uso do conjunto *crowdsourcing* e aprendizado de máquina. Guo (2015), por exemplo, usa a inteligência humana com uma função de similaridade para *clusterizar* imagens. Na mesma direção, Wang (2012) usa uma abordagem humano-máquina para encontrar diferentes registros em bancos de dados que se referem a uma mesma entidade. Neste caso, o esforço da *crowd* é reduzido, pois ela trabalha sobre os resultados de saída obtidos pelo processamento da máquina.

Em todas as abordagens híbridas, existe a necessidade de que métodos automáticos sejam aplicados sobre os conteúdos analisados, sendo que a maior parte destes métodos se baseia na extração de características obtidas pelo processamento destes conteúdos.

No capítulo 7, é apresentado um experimento que faz uso deste conceito de método híbrido, combinando a análise de áudio com a proposta de uso de computação humana.

2.5.CONSIDERAÇÕES

Este capítulo apresentou os vídeos gerados pelo usuário (*UGVs*) e suas características particulares. Estes vídeos são o alvo na proposta de sincronização. Além disso, foi apresentado um modelo de sincronização capaz de sincronizar os vídeos gerados por usuários. Este modelo, porém, depende de um método capaz de processar os vídeos encontrado a sincronização entre eles. Desta forma, também foram apresentados métodos para sincronização de vídeos, tanto tradicionais quanto o proposto por esta tese.

Diversos estudos foram realizados para permitir o desenvolvimento do alinhamento temporal. A partir destes estudos, foi possível compreender que para os usuários terem a sensação de que um conjunto de vídeos, com conteúdos correlacionados, estejam sincronizados não implica necessariamente que eles utilizem uma referência temporal única, mas que as apresentações de seus *frames* correlacionados sejam realizadas em um intervalo de tempo aceitável ao usuário.

Com isto em mente e levando em consideração o modelo (*Edit-Share-Control*), onde o público participa cada vez mais do ciclo de vida dos conteúdos gerados, surgiu a concepção de que os usuários poderiam atuar no processo de sincronização de vídeos. De fato, este processo é compatível tanto com a ideia de alinhamento temporal, quanto com a ideia de se identificar os pontos de sincronização entre os conteúdos a serem sincronizados.

3. CROWDSOURCING

O capítulo anterior apresentou um método de sincronização entre vídeos gerados por usuários, através do alinhamento temporal dos seus conteúdos. A abordagem oferece suporte para a sincronização de diversos conteúdos distribuídos em vários provedores para diversos clientes, gerando uma apresentação sincronizada localmente para cada cliente. Porém, a parte do processamento e da geração dos pontos de sincronização entre os conteúdos ainda deve ser explorada.

Alguns dos métodos citados na seção 2.4.4 são dependentes das características do processo de captura (câmeras estáticas, foco exclusivo em objetos capturados, marcações de tempo e a necessidade de profissionais envolvidos durante o processo). Em vídeos capturados por usuários, no entanto, como não há nenhum tipo de controle da captura cada usuário registra o que deseja, quando e durante o tempo que for necessário, resultando em vídeos com características bastante heterogêneas. Desta forma, esta tese propõe uma nova forma de processar e sincronizar os vídeos, tendo como base a percepção humana e da heterogeneidade dos conteúdos dos *UGVs*.

O capítulo traz uma breve introdução sobre o uso de pessoas em processos computacionais, direcionando conceitos de computação humana e, posteriormente, de *crowdsourcing* para o processamento dos pontos de sincronização entre vídeos.

3.1. COMPUTAÇÃO HUMANA

O paradigma denominado Computação Humana (do inglês *Human Computation*) está associado à solução de problemas como os anteriores, identificando quais tarefas podem ser automatizadas e quais requerem processamento humano (LAW e AHN, 2011). Para maximizar os benefícios do uso da computação humana, deve-se direcionar o esforço dos humanos apenas em tarefas que realmente requerem a sua atenção. Law & Ahn (2011) as denominam de Tarefas para Inteligência Humana (do inglês *Human Intelligence Task – HIT*). Projetos de sucesso conhecidos e que são exemplos do uso da computação humana são o *reCAPTCHA* (VON AHN, MAURER, *et al.*, 2008), o *ESP Game* (STEPHEN ROBERTSON, 2009) e o *Duolingo* (AHN, 2011).

O *ESP Game* é um sistema modelado na forma de um jogo *on-line*, cujo objetivo é coletar anotações em imagens, que são apresentadas aleatoriamente aos usuários. O jogo permite criar um *dataset* de imagens anotadas com vários rótulos descritos por seres humanos, que computadores ainda trabalham com dificuldade para conseguir, devido à diversidade das imagens, oclusão nas fotos, e necessidade de interpretação de contexto ou inferência subjetiva. O *Google* posteriormente adquiriu a licença do sistema e a utilizou como base para sua própria versão, o *Google Image Labeler*.

O *CAPTCHA* (*Completely Automated Public Turing test to tell Computers and Humans Apart*) testa se uma ação está sendo feita por um usuário humano ou por algum tipo de robô (VON AHN, MAURER, *et al.*, 2008). Com base nesta premissa uma pequena tarefa que requer inteligência humana para ser concluída foi incorporada ao sistema original. Esta ideia originou o *reCAPTCHA*, que usa a estratégia proposta pelos testes *CAPTCHA*, de uma forma que a inteligência humana possa ser usada para apoiar aplicações como o reconhecimento de texto em situações onde os sistemas automáticos apresentam dificuldades (VON AHN, MAURER, *et al.*, 2008).

O *Duolingo* tem como um de seus objetivos a tradução de textos com o uso da inteligência humana. A ideia base é que os usuários traduzem textos enquanto aprendem uma nova língua (AHN, 2011). A tradução é outro problema que, apesar das numerosas técnicas automáticas, exige o uso da inteligência humana para gerar um resultado com qualidade relevante (BYWOOD, 2017). A tradução automática enfrenta dificuldades relacionadas a especificidades de um domínio, uso de gírias, expressões linguísticas, regionalismos, entre outros.

Em suma, usando um trecho de (LAW e AHN, 2011):

> *"... human computation is simply computation that is carried out by humans. Likewise, human computation systems can be defined as intelligent systems that organize humans to carry out the process of computation whether it be performing the basic operations, taking charge of the control process itself, or even synthesizing the program itself. The meaning of "basic" varies, depending on the particular context and application. For example, the basic unit of computation in the calculation of a mathematical expression can be simple operations or composite operations that consist of several simple operations. On the other hand, for a crowd-driven image labeling system, a user who generated a tag that describes the given image can also be considered to have performed a "basic" unit of computation."*

3.2.WINSDOM OF THE CROWD

Basicamente, a computação humana propõe que seres humanos sejam utilizados para o processamento no lugar de um sistema computacional. Esse é um conceito muito abrangente, permitindo a utilização da computação humana em diversos contextos. O conceito de *crowdsourcing* está intimamente ligado à ideia de computação humana, recrutando pessoas para realizar tarefas computacionalmente complexas e agrupando estas contribuições em uma solução única.

Francis Galton escreveu em 1907 o artigo *Vox Populi* (GALTON, 1907), em que relatou uma experiência onde um conjunto de pessoas em uma feira agrícola tentou adivinhar o peso de um animal. Galton verificou que a média de todos os palpites presumidos convergiu para um valor muito próximo ao peso real do animal. A análise dos valores fundamentou o que ele chamou de *Winsdom of the Crowd*, de acordo com o qual um grupo heterogêneo grande o suficiente tende a fornecer um resultado tão bom quanto um perito. Em suma, um grupo de pessoas trabalhando de forma cooperativa pode solucionar problemas complexos a partir da agregação das contribuições individuais e independentes de cada membro do grupo.

A Inteligência Coletiva, outro conceito fortemente relacionado à ideia de computação humana, é aquela que emerge a partir dos diferentes conhecimentos compartilhados pelas pessoas sobre um mesmo assunto. Pode-se afirmar que ninguém tem todo o conhecimento sobre um assunto, mas que todos têm algum conhecimento sobre ele e que a combinação dos saberes individuais gera a inteligência coletiva.

De acordo com Levy (1993):

"A inteligência coletiva é uma inteligência distribuída, incessantemente valorizada, coordenada em tempo real, resultando em mobilização efetiva de competências, que busca reconhecimento e enriquecimento de pessoas."

O conceito de inteligência coletiva foi criado a partir de discussões por (LÉVY, 1993), relacionadas com tecnologias da inteligência. Hoje em dia, a Internet tem sido usada como uma ferramenta para tornar mais ágil este tipo de inteligência coletiva e assim o conceito tem obtido novos contornos.

Existem três maneiras de se gerar a inteligência coletiva: inconsciente, consciente e plena (CAVALCANTI e NEPOMUCENO, 2017). Na inconsciente, o usuário realiza contribuições sem saber, simplesmente deixando rastros de suas ações, como a navegação em um *site*, através do preenchimento de um formulário, ou ao clicar em um *link*. Na consciente, os usuários envolvidos realizam contribuições conscientes para alcançar a inteligência. Exemplos desta inteligência são o desenvolvimento de softwares livres e o suporte de usuários em fóruns nos quais as pessoas propõem problemas a serem resolvidos. Já a inteligência plena, envolve tanto a inconsciente quanto a consciente em um único ambiente.

Tanto o conceito de *Winsdom of the Crowd* quanto o de Inteligência Coletiva são de grande valor para o entendimento de *crowdsourcing*, que se utiliza do conhecimento e contribuições geradas por pessoas para a solução de problemas.

Crowdsourcing é uma abordagem cooperativa que usa a sabedoria da multidão (*Winsdom of the Crowd*) para oferecer resultados de boa qualidade usando contribuições de um grupo (*crowd*) de colaboradores. Uma abordagem *crowdsourcing* deve ser capaz de distribuir, coletar, validar e mesclar grandes quantidades de contribuições (JISUP HONG, 2011) (HAAS, 2015) (MO, 2013). Uma vez que o *crowdsourcing* foi projetado para lidar com um vasto número de colaboradores e contribuições para tarefas que requerem inteligência humana (HOWE, 2006), ele é apropriado para permitir que o paradigma da computação humana seja aplicado em um cenário de cooperação *online*, aumentando o desempenho do sistema através de tarefas paralelas (ROHWER, 2010), e melhorando a exatidão dos resultados gerados de acordo com o conhecimento da *crowd*.

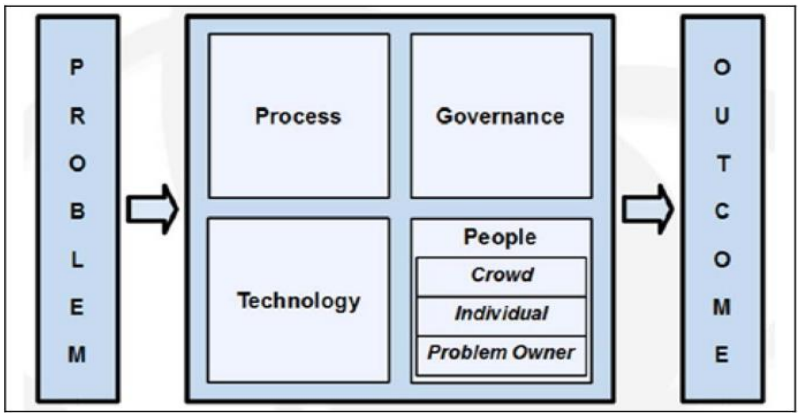
Segundo Estelles e Gonzáles (2012):

> *"Crowdsourcing is a type of participative online activity in which an individual, an institution, a non-profit organization, or company proposes to a group of individuals of varying knowledge, heterogeneity, and number, via a flexible open call, the voluntary undertaking of a task. The undertaking of the task, of variable complexity and modularity, and in which the crowd should participate bringing their work, money, knowledge and/or experience, always entails mutual benefit. The user will receive the satisfaction of a given type of need, be it economic, social recognition, self-esteem, or the development of individual skills, while the crowd-sourcer will obtain and utilize to their advantage what the user has brought to the venture, whose form will depend on the type of activity undertaken."*

A abordagem *crowdsourcing* se apoia sobre quatro pilares fundamentais: a tarefa a ser realizada (*task*), o dono do problema a ser resolvido (*crowdsourcer* ou

owner), a multidão (*crowd*), e a plataforma de *crowdsourcing* (HOSSEINI, 2014). O *crowdsourcer* define o problema a ser resolvido. Utilizando uma plataforma e outras ferramentas ele define o processo, criando as *tasks* que deverão ser executas pela *crowd* e podendo fazer seu gerenciamento (acompanhando as *tasks* realizadas e resultados). Ao final, um produto (*outcome*) é entregue com a provável solução do problema. A Figura 5, extraída de (PEDERSEN, 2013), ilustra o modelo conceitual deste processo.

Figura 5 - Modelo conceitual de *Crowdsourcing*



Fonte: (PEDERSEN, 2013)

A *task* é projetada para que um membro da *crowd* (*worker*) gere uma contribuição que possa ajudar na solução do problema. Várias instâncias de uma mesma *task* podem ser geradas e estas instâncias são apresentadas aos *workers* para serem executadas (DIFALLAH, 2015).

O *crowdsourcer* é o proprietário de um projeto. Pode ser um indivíduo ou instituição que deseja ter um problema resolvido. Ele é responsável por iniciar o processo de *crowdsourcing*, definindo quais *tasks* devem ser concluídas e como devem ser apresentadas aos *workers* (HOSSEINI, 2014).

A *crowd* é a força de trabalho que move o processo. Cada *worker* executa seu trabalho de forma independente, de modo que as instâncias das tarefas podem ser executadas em paralelo.

A plataforma de *crowdsourcing* é o centro de todo o processo. É um ponto de entrada para o *crowdsourcer* que torna as tarefas disponíveis aos *workers*. Este tipo de ambiente pode ser sofisticado como o *Amazon Mechanical Turk* (*mturk.com*) e o *MicroWorkers (microworkers.com)*, ou ainda, um sistema construído pelo próprio *crowdsourcer*. O uso de uma plataforma de *crowdsourcing* comercial traz benefícios tais como: ferramentas de gerenciamento dos *workers*, *tasks* e resultados das

contribuições. Por outro lado, a construção de uma plataforma própria permite um alto nível de personalização da aplicação.

O esforço requerido para a realização de uma e forma pela qual a *crowd* deve contribuir trazendo o seu trabalho, dinheiro, conhecimento e/ou experiência, sempre implica em benefício mútuo. O *worker* receberá um tipo particular de recompensa (pagamento, satisfação social, autoestima, ou o desenvolvimento de habilidades individuais), enquanto o *crowdsourcer* vai obter e usar a contribuição que o *worker* trouxe para a sua solução.

Outros conceitos fundamentais ao entendimento do processo são: o de contribuição, que é o resultado da execução da tarefa (*task*) por um *worker*; o de distribuição, que é uma funcionalidade assumida pela plataforma, que dita quais tarefas devem ser realizadas por quais *workers*; e o de *workflow*, que é o planejamento de como as *tasks* devem ser pensadas, considerando possíveis dependências entre atividades.

Segundo (DOAN, RAMAKRISHNAN e HALEVY, 2011), a resolução de problemas de propósito geral usando *crowdsourcing* impõe quatro desafios fundamentais a serem enfrentados:

1. Como recrutar e reter a *crowd*?

   Sem membros ativos para executar as *tasks*, nenhum problema poderia ser resolvido por *crowdsourcing*. Desta forma são necessárias estratégias para convocar membros para a *crowd* e mantê-los ativos na execução das *tasks* através de incentivos (JOHN P. RULA, 2014) (L. PU, 2016);

2. Quais contribuições a *crowd* pode fornecer?

   Cada problema a ser resolvido via *crowdsourcing* pode requerer estratégias diferentes para a solução. Assim, é necessário definir quais tipos de tarefas serão criadas de forma à *crowd* executar estas tarefas e ao final, solucionar o problema;

3. Como combinar e usar as contribuições feitas para resolver o problema alvo?

   Um problema a ser resolvido pela *crowd* pode gerar de dezenas a milhares de contribuições. O desafio passa a ser como agregar todas estas contribuições de forma a gerar uma resposta ao problema inicial (HUNG e AL., 2013);

4. Como avaliar as contribuições e a própria *crowd*?

Com o uso de técnicas automáticas, muitas vezes é possível se prever o correto funcionamento do algoritmo e da máquina, por meio de análise de suas limitações e execução de testes. Porém, em *crowdsourcing*, como pessoas estão envolvidas no processo, a solução fica suscetível aos seus erros, seja por causa de falhas na execução das tarefas, seja devido à má intenção do membro da *crowd* (GARDLO, EGGER, *et al.*, 2014);

3.3.*CROWDSOURCING* NO PROCESSAMENTO DE VÍDEOS

O uso de técnicas de *crowdsourcing* para processamento de vídeos não é uma proposta nova, e pode ser encontrada em múltiplos trabalhos como mostrado a seguir. Com o intuito de compreender como estes trabalhos utilizam *crowdsourcing* em seus contextos e verificar se já existia literatura semelhante que utilizasse a *crowd* na sincronização de vídeos, foi realizada uma revisão bibliográfica sistemática. Como resultado, foram encontrados diversos trabalhos que utilizam *crowdsourcing* nos mais diversos aspectos de processamento de vídeos: anotação, distribuição, geração, sincronização, avaliação de qualidade, recuperação, sumarização e recomendação.

3.3.1. Design do Estudo

O processo de revisão sistemática da literatura apresentado nesta seção atende às diretrizes propostas por Dyba *et al.* (2007), e Razavian & Lago (2015). O protocolo de revisão utilizado e o método usado para executar o processo de pesquisa, e os dados extraídos são apresentados a seguir.

A revisão teve como objetivo caracterizar os sistemas *crowdsourcing* que utilizam a *crowd* para processamento de vídeos. O primeiro passo do processo para tal foi a escolha das palavras-chave para a pesquisa. Com base nas questões propostas e no contexto da pesquisa, duas palavras foram definidas: *crowdsourcing* e vídeo. Além disso, os termos alternativos para *crowdsourcing* foram adicionados, chegando a seguinte *string* de busca:

(video AND
(crowdsourcing OR crowd-based OR crowdsourced OR crowdsource))

Para realização da busca foram utilizadas as seguintes bibliotecas digitais como fonte: *ACM Digital Library*, *Engineering Village*, *IEEE Xplore*, *Science Direct* e *Scopus*. Elas foram escolhidas devido a sua importância dentro da comunidade científica e o fato de que seu índice era acessível durante a realização do trabalho.

Na busca, um estudo se torna um candidato a trabalho selecionado se ele contém os termos da *string* de pesquisa em seu resumo, título ou palavras-chave. Além disso, uma pesquisa manual nos *proceedings* do ACM *CrowdMM* foi conduzida para certificar que todos os documentos relacionados a partir deste evento estariam incluídos nos resultados. Depois de concluir a pesquisa, verificou-se que todos os documentos relevantes estavam inclusos na busca automática.

Ao final da primeira etapa de busca, 1430 publicações foram identificadas como possíveis trabalhos resultantes para a revisão. Destes, 1217 foram excluídos aplicando critérios de exclusão com a leitura de seus títulos e resumos, como por exemplo artigos que não abordavam o uso de vídeos, mas apenas outras mídias. Também foram excluídos artigos duplicados e outros documentos que não eram artigos (chamadas de trabalho e descrição de *proceedings* são exemplos).

Em seguida, um segundo filtro de inclusão e exclusão foi aplicado, mas desta vez através da leitura dos trabalhos em sua integra, resultando na seleção de 119 artigos. Porém, durante a leitura foi identificado um alto grau de similaridade entre alguns artigos, fazendo com que fossem verificadas possíveis duplicações de trabalhos. Após esta verificação, foram obtidos 106 estudos primários para execução do mapeamento.

O principal resultado desta revisão para o contexto desta tese é a caracterização de como a *crowd* é utilizada no processamento de vídeos. Sendo assim, apenas este resultado será reportado neste capítulo. Entretanto, um detalhamento mais preciso de todo o processo de revisão será apresentado em (SEGUNDO, SANTOS e DE AMORIM, 2016).

3.3.2. Uso da *Crowd*

O processo de revisão sistemática revelou que o uso da *crowd* para processamento de vídeos está associado, essencialmente, às seguintes atividades: anotação, distribuição, geração, agrupamento, apresentação, avaliação de

Qualidade de Experiência (QoE) e Qualidade de Serviço (QoS), recomendação, sumarização e sincronização de vídeos.

A seguir, os principais trabalhos reportando o uso de *crowdsourcing* nas atividades listadas são discutidos.

3.3.2.1. Anotação de Vídeos

Anotações são normalmente usadas para atribuir informações sobre um recurso ou associar diferentes recursos. Neste contexto a *crowd* é utilizada para fazer anotações e resolver diversos tipos de problemas. Estes trabalhos incluem: anotar o grau de criatividade de vídeos disponibilizados na Internet para classificá-los e definir quais deles são considerados criativos (REDI, OḤARE, *et al.*, 2014); criar legendas para vídeos educacionais (DESHPANDE, TUNA, *et al.*, 2014) e vídeos em geral, utilizando elementos de jogos para incentivar a participação (KACORRI, SHINKAWA e SAITO, 2014); anotação de palavras-chave sobre os vídeos (FERRACANI, PEZZATINI, *et al.*, 2015) para auxiliar a busca e recuperação de conteúdo; identificação de acontecimentos em vídeo de vigilância (GADGIL, TAHBOUB, *et al.*, 2014); descrever ações humanas, identificação de personalidade e interpretação de emoções para treinamento de técnicas automáticas, classificação de vídeos e métodos de busca (TAHBOUB, GADGIL, *et al.*, 2015) (ARAN, BIEL e GATICA-PEREZ, 2014) (RIEK, O'CONNOR e ROBINSON, 2011) (SANCHEZ-CORTES, KUMANO, *et al.*, 2015) (BURMANIA, PARTHASARATHY e BUSSO, 2016) (BAVEYE, DELLANDREA, *et al.*, 2015) (SPIRO, 2012) (BIEL e GATICA-PEREZ, 2013) (PARK, SHOEMARK e MORENCY, 2014); anotações no formato de áudio para descrição dos vídeos (LASECKI, THIHA, *et al.*, 2013); anotações semânticas (SULSER, GIANGRECO e SCHULDT, 2014) em eventos esportivos e anotações para sumarização dos mesmos (TANG e BORING, 2012); anotações de posicionamento geo-espacial para reconstrução de ambientes (CHEN, LI, *et al.*, 2015) (GOTTLIEB, CHOI, *et al.*, 2012); anotações com marcações temporais para pontos de interesse específicos na *timeline* do vídeo (WU, ZHONG, *et al.*, 2014) (XU e LARSON, 2014); marcações de eventos de interesse do público no vídeo (STEINER, VERBORGH, *et al.*, 2011) (CHORIANOPOULOS, 2012) (VLIEGENDHART, LONI, *et al.*, 2013) (CRAGGS, KILGALLON SCOTT e ALEXANDER, 2014) (WU, THAWONMAS e CHEN, 2011) (BHIMANI, NAKAKURA,

et al., 2013); rótulos sobre a qualidade do vídeo (FREIBURG, KAMPS e SNOEK, 2011) (HAN e LEE, 2014); identificação de objetos (TANG e BORING, 2012) (BERTINI, DEL BIMBO, *et al.*, 2013) (PINTO e VIANA, 2013); anotação sobre acontecimentos em ninhos de pássaros (DESELL, GOEHNER, *et al.*, 2015); e recuperação de informações (SANTOS-NETO, PONTES, *et al.*, 2014).

3.3.2.2. Distribuição de Vídeos

A distribuição de vídeo envolve contribuições sobre como os vídeos podem ser distribuídos utilizando contribuições da *crowd*. Zhou *et al.* (2015) que apresentam um ambiente de distribuição de vídeo baseado em P2P, onde os *workers* podem contribuir fornecendo dispositivos para atuar como pequenos servidores. Neste caso, as contribuições da *crowd* se limitam a “alugar” o dispositivo do usuário e sua conexão. Chen *et al.* (2015) descrevem soluções para problemas de rede e nuvem, bem como uma nuvem-assistida para transmissão ao vivo de vídeos, que é destinada para a apresentação de *UGVs*. Bohez *et al.* (2014) apresenta uma estrutura criada para distribuição, em grande escala, de vídeos gravados usando dispositivos móveis.

3.3.2.3. Geração de Vídeos

Estudos primários com contribuições sobre geração de vídeos trazem *UGVs* no escopo do trabalho. Estes vídeos são gerados através de uma *task* para a *crowd*, ou obtidos a partir de repositórios Web, como o YouTube ou o Hulu.

Ferracani *et al.* (2015) usam a *crowd* para gerar vídeos. Os usuários são convidados a fazer *upload* de seus vídeos favoritos, adicionar *tags*, e classifica-los. Ao final, é apresentado um sistema para executar a recomendação de vídeo baseada em item através do conteúdo fornecido pela *crowd* e anotações automáticas.

Wang *et al.* (2014) e Zhang *et al.* (2015) utilizam vídeos gerados pela *crowd* no intuito de extrair *keyframes* para o mapeamento 3D de um objeto; Chen *et al.* (2015) usam o conteúdo gerado pelo usuário para mapear espaços comuns. Eles descrevem o processo de criação de um mapa usando o conteúdo fornecido pelo usuário, usando inúmeros algoritmos para projetar o espaço no mapa; Göransson e

Jensfelt (2013) usam vídeos em RGB-D criados pela *crowd* no processo de mapeamento 3D de objetos.

Chen *et al.* (2015) e Zhang *et al.* (2015) utilizam soluções que permitem uma estrutura de distribuição de vídeos para conteúdos ao vivo gerados pelos usuários.

Tag *et al.* (2014) e Bhimani *et al.* (2013) usam a *crowd* para contribuir em processos de criação de vídeos. As pessoas assistem, votam e filmam, com o objetivo de juntos construírem um novo vídeo; para criar um *dataset* de vídeos com simulação de emoções, Spiro (2012) coletou *UGVs* através da interface de um jogo. Neste jogo, os jogadores têm de assistir a um vídeo curto com ações que estão sendo executadas por outra pessoa, e em seguida, tentar imitá-lo. Os jogadores também são incentivados a enviar seus desempenhos para os amigos, espalhando o jogo no processo e conseguindo mais membros e vídeos.

Venkatagiri (2015), aborda como fazer a prevenção de custos desnecessários com upload de *UGVs*. Usando a análise do conteúdo para verificar se o envio daquele vídeo irá trazer ganho de conteúdo ao usuário final, ocorre a redução dos custos associados à rede e energia. Um algoritmo é executado no cliente, enviando metadados para o servidor. Usando esses metadados e as consultas feitas por usuários, os vídeos que serão enviados ao servidor são escolhidos.

Wilk *et al.* (2015) faz com que a *crowd* transmita fluxos de vídeo ao vivo para um servidor. Os melhores vídeos em relação a qualidade de imagem e áudio, são selecionados por outros membros da *crowd*, criando uma nova composição para ser apresentada aos novos usuários.

O uso do YouTube é uma alternativa para adquirir *UGVs* sem necessidade de criar *tasks* para isto. São exemplos desta abordagem os trabalhos de Huberman *et al.* (2009), e Kato *et al.* (2015), que além dos vídeos, usam metadados e informações do vídeo para seu processamento. Frey e Antone (2013) também utilizam vídeos provenientes do YouTube, mas em seu contexto eles tentam analisar o conteúdo e encontrar eventos espaciais e temporais entre câmeras usando sinais de áudio e objetos em vídeo. Ao combinar vídeos, é possível ver um evento de diferentes ângulos.

3.3.2.4. Sincronização de Vídeos

Dentre os estudos primários selecionados, alguns apresentam contribuições para a apresentação síncrona com vídeos usando contribuições na criação de âncoras e links que conectam vídeos e outras mídias, espacialmente ou contextualmente, mas nenhum utilizou a crowd para criar elos de sincronização entre múltiplos vídeos.

Usando redes sociais, alguns artigos sincronizam vídeos com outros tipos de mídias. Kato *et al.* (2015) utilizam serviços de redes sociais para estruturar as relações semânticas cronológicas entre vídeos de notícias; Venkatagiri *et al.* (2015) utilizam informações espaciais e temporais para relacionar *UGVs*, então quando um usuário solicita um vídeo, o vídeo de melhor qualidade avaliado pelos demais usuários é disponibilizado; Bertini *et al.* (2013), Vliegendhart *et al.* (2013), e Ordelman *et al.* (2015) utilizam ferramentas de anotação onde os usuários podem comentar vídeos a nível de frames e adicionar recursos de Facebook e Wikipedia para complementar a apresentação dos vídeos.

Informações espaciais dos vídeos (informações de GPS e análise de frames) são utilizados para fornecer pontos de sincronização espacial. Jain *et al.* (2013), Zhang *et al.* (2015), e Carlier *et al.* (2014) utilizam estas informações para reconstrução de cenas e objetos 3D.

3.3.2.5. Avaliação de Qualidade de Vídeos

A QoE é a medida da experiência de um usuário com um serviço, neste caso, vídeos. A avaliação subjetiva é um método comum de avaliação da qualidade. Como os testes subjetivos podem ser demorados e caros, as soluções que usam a *crowd* mostram ser uma alternativa interessante ao caso. A QoS está relacionada a medição, melhoria e garantia antecipada de taxas de transmissão, de erro, outras características quantitativas.

Diversos tipos de vídeos têm sido usados nas avaliações de qualidade dos trabalhos encontrados. Rainer e Kimmerer (2014), Egger *et al.* (2014), e Shahid *et al.* (2014) avaliam o uso de *streamings* de vídeo adaptativos, avaliando o impacto da mudança de qualidade para os usuários e os melhores valores de parâmetros para a transmissão; serviços de streaming como o YouTube, também são avaliados

utilizando as técnicas de *crowdsourcing* (HOßFELD, SEUFERT, *et al.*, 2011). Não apenas vídeos são avaliados, Rainer *et al.* (2015) apresentam um experimento para medir o impacto do atraso na sincronização entre destinos em vídeos sociais.

Práticas e ferramentas que podem ser utilizadas na avaliação de vídeo usando *crowdsourcing* fazem parte da contribuição dos estudos encontrados: as melhores práticas para teste de QoE são propostas por HOßFELD *et al.* (2014); Keimel *et al.* (2012) apresentam os desafios de avaliação de qualidade de vídeo utilizando *crowds*; um framework para a análise de avaliações de qualidade visual usando *crowdsourcing* é apresentado por Fremuth *et al.* (2015); enquanto que Pauliks *et al.* (2013), e Zegarra *et al.* (2015) apresentam as métricas a serem utilizadas na avaliação; e Anegekuh *et al.* (2014) apresentam metodologias para identificar contribuições de baixa qualidade.

3.3.2.6. Recuperação de Vídeos

Diversos trabalhos propõem o uso da *crowd* para melhorar os métodos de recuperação de vídeos a partir de informações do seu conteúdo. Apesar da atividade de anotação, já abordada, estar intimamente ligada à questão da recuperação de conteúdos, os trabalhos discutidos nessa subseção estão ligados ao uso explícito de *crowdsourcing* para a recuperação de vídeos.

Sulser *et al.* (2014) utilizaram anotações de vídeo como base para um sistema de recuperação que permite aos usuários pesquisar sequências de vídeos com base em uma especificação gráfica do comportamento das equipes, ou do movimento individual do jogador em vídeos de jogos esportivos; Henter *et al.* (2012) usaram informações fornecidas pelos usuários na sua análise para melhorar o mecanismo de sugestão de *tags* no YouTube, facilitando o armazenamento de informações e recuperação; Santos *et al.* (2014) também apresentaram contribuições para as *tags* do YouTube. Eles perguntaram à *crowd* quais os termos de consulta que eles usariam para procurar um vídeo. Os termos supostamente melhorariam a recuperação de vídeo pelo motor de busca do YouTube. No lugar de *tags*, Vasudevan *et al.* (2013) utilizam *twitters* para identificar acontecimentos em jogos de futebol e assim recuperar momentos importantes dos jogos quando necessário.

3.3.2.7. Sumarização de Vídeos

A sumarização de vídeo é o processo de redução de um vídeo usando algoritmos, para criar um resumo que contém os pontos mais importantes do vídeo original. Usando *crowdsourcing*, os autores dos estudos acreditam que os melhores resultados podem ser gerados pela *crowd* ou que a mesma pode ser usada para validar resultados produzidos por métodos computacionais.

Kim *et al.* (2014) tiveram como objetivo validar um algoritmo de sumarização automático, criando histórias baseadas em vídeos do YouTube e imagens do Flickr. Eles usaram *crowdsourcing* como método de avaliação, pedindo aos usuários para selecionar frames que resumem o vídeo e comparar com os frames selecionados usando o algoritmo alvo; Khosla *et al.* (2013) também usaram a *crowd* para validar um algoritmo de sumarização próprio. Eles usaram um grupo de especialistas para avaliar um conjunto de vídeos sumarizados automaticamente e comparou-os com a sumarização coletada da *crowd*; Wu *et al.* (2011) criaram uma estrutura baseada no uso da *crowd* para gerar resumos. Os *workers* só precisavam clicar em uma tecla toda vez que acreditavam que o vídeo estava apresentando um ponto de destaque.

3.3.2.8. Recomendação de Vídeos

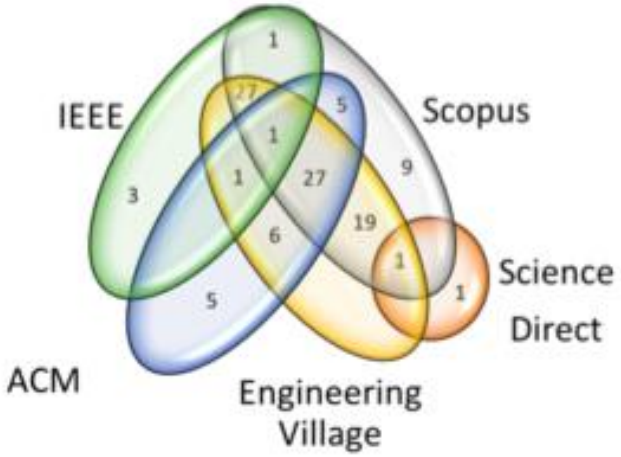
Sistemas de recomendação estudam padrões de comportamento para saber o que alguém vai preferir, entre uma coleção de itens que ele nunca experimentou. Eles procuram prever a preferência que um usuário teria com um item, no caso específico, um vídeo. Ferracani *et al.* (2015) apresentaram um sistema que executa a recomendação de vídeo baseada em item, usando uma descrição baseada em conteúdo de vídeos obtidos de anotações automáticas e *crowdsourced*; Bertini *et al.* (2013) usaram um sistema em *piggybacking* que consumia dados de outras plataformas como o *Twitter* e construiu um *framework* que combinou um módulo de criação de perfil e análise de atividade dos usuários para gerar recomendações; Vasudevan *et al.* (2013) também utilizaram redes sociais (ex. Twitter) para recomendar vídeos. A partir de *tags* extraídas dos posts do Twitter conseguiram recomendar vídeos com comentários relacionadas àquelas *tags*.

3.3.3. Dados do mapeamento

Na seção anterior foi apresentado um resumo dos artigos que fizeram parte dos estudos primários do mapeamento, mostrando as atividades nas quais a *crowd* é utilizada no processamento de vídeos. Nesta seção serão apresentados dados quantitativos que resumem alguns dados quantitativos extraídos do mapeamento.

A Figura 6 mostra um diagrama de *Venn* representado pelos grupos de artigos originários de cada biblioteca utilizada, e em suas intersecções os artigos que aparecem em mais de uma biblioteca. O maior grupo é o proveniente do *Scopus*, que possuí não só o maior número de artigos, mas também a maior quantidade de artigos exclusivos a uma das bibliotecas.

Figura 6 - Fontes dos Artigos Selecionados



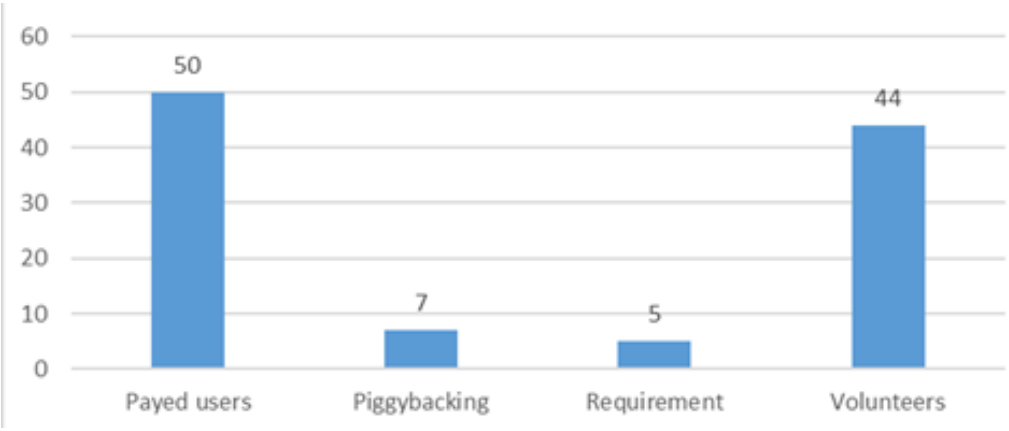
Fonte: Elaborada pelo autor

Destes 106 estudos primários selecionados, foi extraído como o recrutamento da crowd em cada um foi realizado (Figura 7): 50 recrutaram os *workers* através de pagamentos, 44 utilizaram voluntários, 7 usaram sistemas em *piggibacking* e 5 exigiram a contribuição do usuário (empregadores, estudantes de laboratório e outros). Os estudos que fizeram recrutamento através de pagamentos utilizaram uma plataforma de *crowdsourcing* online para realizar esta tarefa. Dentre as plataformas utilizadas, estão: *Amazon Mechanical Turk* (32 artigos), *Microworkers* (10 artigos), *CrowdFlower* (7 artigos), *QualityCrowd* (2 artigos), *Clickworker* (um artigo) e *CrowdMOS* (um artigo).

O pagamento realizado aos *workers* variou de acordo com a tarefa a ser realizada. Baveye et al. (2015) pagou US$ 0,05 para os *workers*, que precisavam comparar cinco pares de vídeos selecionando aquele que expressasse a "emoção

mais positiva" ou "a emoção mais calma". Vondrick et al. (2013) também remunerou os *workers* com o valor de US $ 0,05 por objeto anotado, onde uma anotação completa de um vídeo clipe chegou a custar entre US$ 1 e US$ 2. No entanto, o valor médio de uma *task* foi entre: 0,2 e 0,3 dólares.

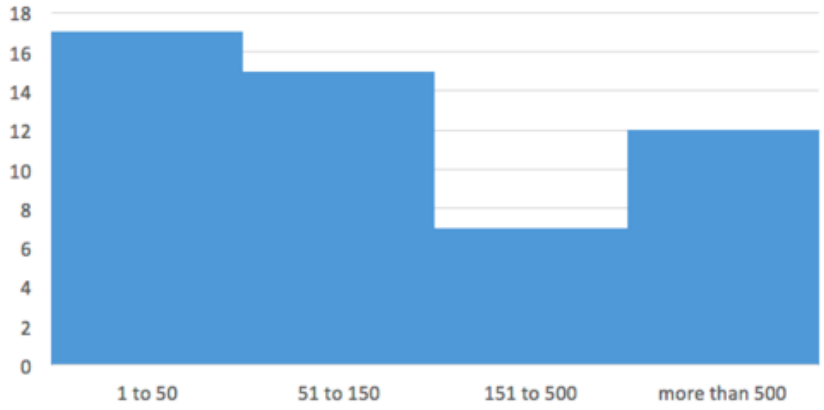
Figura 7 - Forma de recrutamento da *Crowd*



Fonte: Elaborada pelo autor

Informações acerca do número de contribuições realizadas em cada experimento também foram coletadas (Figura 8): de 9 a mais de meio milhão. O recebimento de mais de 500.000 contribuições é uma exceção aos demais artigos, sendo alcançado devido ao uso de um plugin junto ao player do YouTube para coleta de dados (HUBERMAN, ROMERO e WU, 2009), sendo que em geral o número de contribuições é menor que 150.

Figura 8 - Número de Contribuições Coletadas

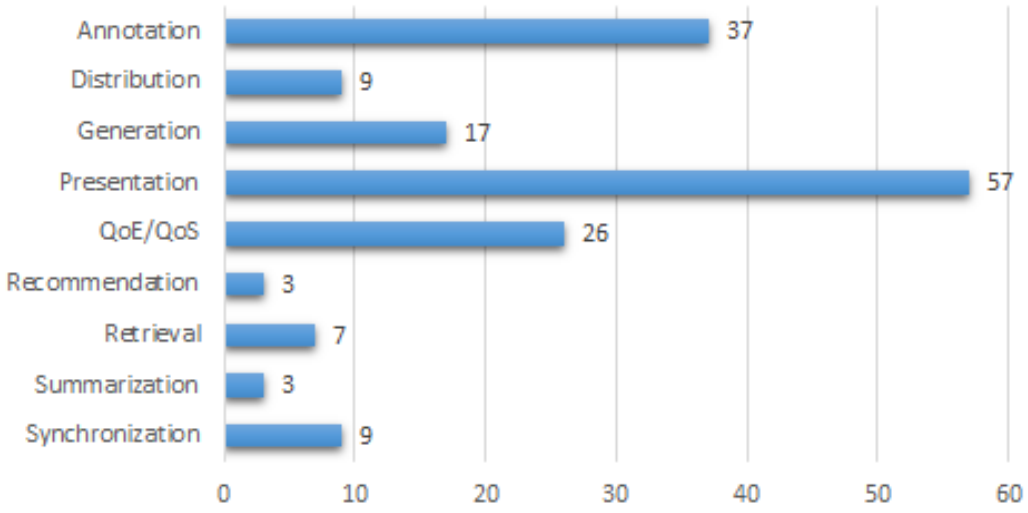


Fonte: Elaborada pelo autor

Por fim, a Figura 9 mostra como os estudos primários foram distribuídos, considerando atividades realizadas pela *crowd* no processamento dos vídeos. As

principais atividades realizadas foram: anotação, criação, análise de qualidade e apresentação dos vídeos.

Figura 9 - Número de Artigos por Atividade



Fonte: Elaborada pelo autor

3.4.CONSIDERAÇÕES

Este capítulo discutiu os conceitos de computação humana e os princípios que regem o uso de *crowdsourcing* em diversos cenários, como forma de mostrar que uma *crowd* pode realizar tarefas complexas, como no caso da sincronização de um conjunto de vídeos.

Em seguida, foi apresentado o resultado de um mapeamento sistemático acerca do uso de *crowdsourcing* no processamento de vídeos. Esse estudo revelou os diversos usos da *crowd* para processar vídeos, realizando atividades que variam desde a anotação à avaliação de qualidade destes vídeos.

Nos próximos capítulo, o conhecimento obtido a partir da revisão dos trabalhos tratando do uso de *crowdsourcing* para o processamento de vídeos será utilizado para definir um modelo funcional que busca dar suporte às principais preocupações envolvendo a sincronização de vídeos com o uso da *crowd*.

Por fim, foi possível verificar que a aplicação do conceito de *crowdsourcing* para a sincronização de vídeos ainda não havia sido reportada na literatura, sendo uma contribuição inédita desta tese para o estado-da-arte em computação.

4. SINCRONIZAÇÃO DE VÍDEOS PELA *CROWD*

Esse capítulo apresenta uma abordagem para especificar a sincronização de vídeos gerados por usuários relacionados a um mesmo evento, com a aplicação do conceito de *crowdsourcing*. A abordagem faz uso da habilidade humana para identificar padrões similares entre pares de vídeos, contornando algumas das limitações das técnicas automáticas, cujos resultados são fortemente dependentes dos aspectos técnicos (resolução, iluminação, intermitência, etc.) do conteúdo capturado.

Os capítulos anteriores discutiram os conceitos de computação humana e *crowdsourcing,* e como os seres humanos podem realizar atividades relacionadas ao processamento de vídeos. Alguns trabalhos citados abordaram o problema da sincronização intermídia em situações nas quais textos e imagens são sincronizados aos vídeos para tentar melhorar a compreensão e a acessibilidade deste conteúdo ou para identificar suas mais relevantes partes. Por outro lado, usar *crowdsourcing* para solucionar o problema de sincronizar múltiplos vídeos com conteúdos correlacionados, não foi mencionado por nenhum dos trabalhos encontrados. Para isso, a *crowd* irá realizar a tarefa de encontrar os pontos de sincronização entre os conteúdos vídeos.

4.1. C-SYNC

No capítulo 2 foi introduzido um modelo de sincronização para múltiplos conteúdos distribuídos, o qual é usado no processo de sincronização. No modelo, ilustrado na Figura 4, um método capaz de identificar os pontos de sincronização entre vídeos é implementado pelo *Content Processor* que faz parte do elemento *Coupler* da mesma figura. Neste modelo, a *crowd* irá atuar como um *Content Processor*, gerando os acopladores a partir do uso das habilidades humanas para processar o conteúdo dos vídeos.

4.1.1. Formalização do Problema da Sincronização de UGVs

No escopo desta tese, o problema da sincronização de vídeos se resume a alinhar um conjunto *V* de vídeos correlacionados a um evento social sobre uma linha

temporal comum a todos estes vídeos. Assim, após estarem sincronizados, estes vídeos irão apresentar, simultaneamente, conteúdos similares e correspondentes uma mesma cena capturada do evento.

Considere um par de vídeos v_1 e v_2, capturados por dispositivos diferentes em um mesmo contexto espaço-temporal de um evento. Os vídeos v_1 e v_2 são considerados como sincronizados se, a partir do tempo de apresentação T_i, do *i*-ésimo *frame* de v_1, e T_j, do *j*-ésimo *frame* de v_2, em que os *frames i* e *j* apresentarem a mesma cena capturada num mesmo instante de tempo e codificada em v_1 e v_2.

A escolha do método para permitir a sincronização de vídeos usando *crowdsourcing* pode ser influenciada por diversos fatores. Alguns deles são discutidos a seguir.

4.1.1.1. Similaridade Visual dos Conteúdos

Vídeos com conteúdos muito similares podem confundir os membros da *crowd* que tentam sincronizá-los, mas também podem favorecer técnicas automáticas que buscam pequenas diferenças entre *frames* para identificar pontos de sincronização entre vídeos. Além disso, a execução de tarefas sobre um mesmo conjunto de vídeos pode fazer com que os *workers* adquiram conhecimento sobre este conjunto, reduzindo o tempo para que cenas similares sejam percebidas. Por outro lado, nestas condições, a execução da tarefa pode se tornar mais entediante, já que o *worker* deve assistir várias vezes conteúdos praticamente idênticos a fim procurar a mesma cena em diversos vídeos.

A Figura 10 apresenta quatro categorias de similaridade entre vídeos capturados ao mesmo tempo e relacionados ao mesmo evento e descritas a seguir. Na figura, cada linha representa um vídeo a ser sincronizado (v_i) e os objetos (o_j), os elementos que podem ser usados como referências para que os pontos de sincronização entre os conteúdos dos vídeos sejam encontrados, e suas variações (o'_j) que representa o objeto a partir de outro ponto de vista.

I. Vídeos com conteúdos muito similares: apresentam objetos de cena muito semelhantes nos seus conteúdos, simplesmente deslocados dentro da *timeline* de cada vídeo. Vídeos capturados por câmeras pertencentes a diferentes canais de TV, mas posicionadas lado a lado em um estádio e

filmando uma mesma partida, com ângulos semelhantes, são exemplos desta categoria. Objetos “visualmente semelhantes” aparecem em cena durante todo o tempo de captura destes vídeos;

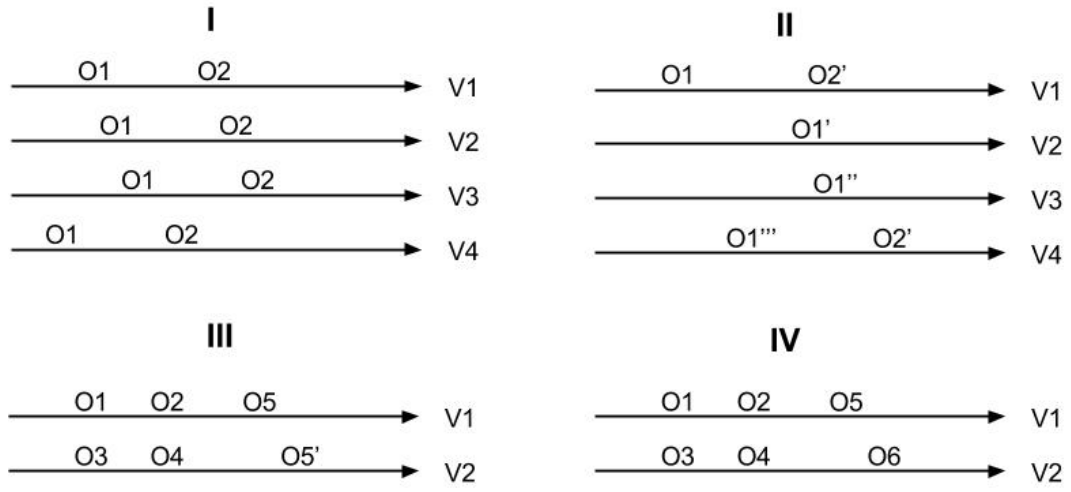
II. Vídeos visualmente diferentes: os objetos de cena capturados nos vídeos são praticamente os mesmos durante toda a duração do conteúdo, no entanto, eles são capturados a partir de diferentes ângulos de visão. No mesmo exemplo anterior, um canal de TV pode oferecer diversas visualizações da mesma cena do jogo, captadas a partir das suas várias câmeras, posicionadas com diferentes ângulos;

III. Vídeos semelhantes por intervalos: os mesmos objetos de cena aparecem nos vídeos, mas apenas durante alguns intervalos, sendo que a maior parte dos conteúdos é visualmente diferente. Isso pode acontecer devido à intermitência na captura, mudanças de ângulo durante a captura, movimentações da câmera, entre outros casos;

IV. Vídeos com similaridade apenas contextual: estão associados a um contexto comum de captura (um mesmo evento social), no entanto, objetos semelhantes são raros ou inexistentes, exigindo o uso de informações que não são facilmente obtidas a partir do processamento do conteúdo visual dos vídeos. Trilhas de áudio ou marcas de conteúdo podem ser usadas para resolver esse problema usando métodos computacionais (SU, 2012) (BANO e CAVALLARO, 2015).

É possível notar que a 1ª categoria é a mais indicada para o uso de processos automáticos de sincronização baseados em similaridade do conteúdo visual. O processamento multimodal, usando, por exemplo, a trilha de áudio ou o fluxo óptico de objetos dos vídeos, (VILMOS ZSOMBORI, 2011) poderia ser usado para encontrar os pontos de sincronização entre vídeos. Entretanto, em qualquer um dos casos, as características do processo de captura vão estar diretamente ligadas à precisão das técnicas automáticas e aos resultados produzidos. Como não é possível garantir tais características para *UGVs*, conforme será demonstrado nos experimentos do Capítulo 6, o desempenho atualmente obtido pelas técnicas automáticas, puramente computacionais, não é suficiente para resolver o problema da sincronização destes vídeos.

Figura 10 - Categorias de similaridade entre objetos de cena



Fonte: Elaborada pelo autor

4.1.1.2. Duração dos Vídeos

Quanto maior a duração do vídeo, mais provável é que alguma parte do seu conteúdo esteja relacionada ao de outros vídeos do conjunto de entrada. Além disso, o vídeo mais longo deste conjunto é um candidato natural para atuar como referência no processo de sincronização, uma vez que possui maior probabilidade de ter parte do conteúdo correlacionada aos outros vídeos a serem processados.

A duração afeta tanto o processamento dos vídeos pela *crowd*, quanto o realizado pelas ferramentas automáticas. Quanto maior a duração do vídeo, mais esforço, em termos de tempo, é exigido na execução da tarefa de buscar um instante particular do seu conteúdo. Além disso, o número de *workers*, tarefas e contribuições da *crowd* (assim como, o número de fases, interações e agregações nos métodos computacionais) será afetado pela duração do vídeo.

4.1.1.3. Continuidade do Conteúdo

O fato do conteúdo de um vídeo não ser contínuo, do ponto de vista semântico, está diretamente relacionado ao seu processo de captura e/ou posterior edição. Em geral, uma descontinuidade semântica no conteúdo de um vídeo está associada a três fatores:

1. Mudança do foco de interesse ou mesmo descuido do usuário durante a captura, resultando em um corte abrupto de cena;

2. Interrupção temporária, de origem intencional ou involuntária, da captura do conteúdo pelo usuário (por exemplo, ativando o botão *pause* da câmera);
3. Inserção de cortes e transições de cenas durante o processo de edição do conteúdo original capturado.

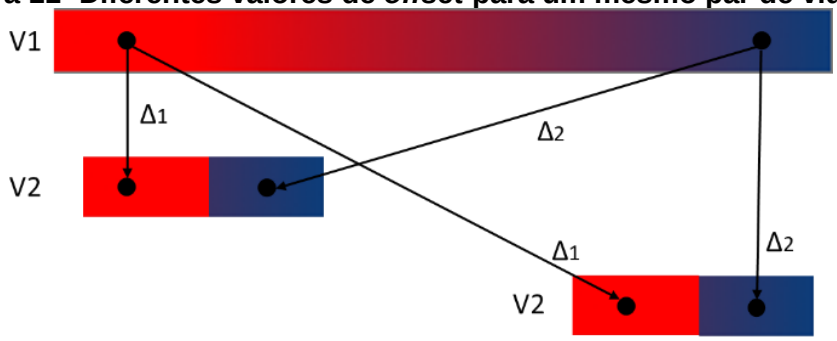
Os fatores (1) e (2) estão associados ao processo de geração dos vídeos, ou seja, na origem destes conteúdos, enquanto que o fator (3) está associado à manipulação dos conteúdos capturados.

As descontinuidades de conteúdo decorrentes de mudanças de cenas (caso 1) podem tornar o processo de sincronização mais difícil, já que partes similares dos conteúdos podem ter sido perdidas exatamente no intervalo em que o plano de captura muda. No entanto, uma mudança de cena pode indicar a ocorrência de um acontecimento particular e de grande interesse durante o processo de captura de um evento (um objeto específico, uma explosão, etc.). Essa ocorrência específica pode ter papel fundamental na definição dos pontos de sincronização entre os vídeos.

Por fim, a inserção de pausas durante a captura dos vídeos e/ou de cortes e transições durante a sua edição (casos 2 e 3) gera um outro problema, que está fora do escopo desta tese. Se um ponto de sincronização é encontrado entre um par de vídeos com conteúdos não necessariamente contínuos, não se pode garantir que seus conteúdos permanecerão sempre sincronizados a partir deste ponto. Isso gera uma interrupção temporária do conteúdo de um vídeo que deslocará sua linha temporal de forma independente dos outros, fazendo com que seu conteúdo pareça dessincronizado após algum tempo de apresentação conjunta dos vídeos.

A Figura 11 mostra o relacionamento entre um vídeo contínuo (V1) e um vídeo com uma descontinuidade (V2). Entre eles não há apenas um offset (Δ) possível. De fato, existem dois valores possíveis: um Δ_1 para os vídeos antes do corte, e um Δ_2 para depois do corte. Esse tipo de situação não é considerada no escopo deste trabalho, já que as estruturas e técnicas foram definidas para os vídeos com conteúdos contínuos e sem edição.

Figura 11- Diferentes valores de *offset* para um mesmo par de vídeos



Fonte: Elaborada pelo autor

4.1.1.4. Conteúdos ao vivo x sob demanda

Quando os *UGVs* são conteúdos transmitidos ao vivo e provenientes de múltiplas fontes, o processo de sincronização irá requerer, a princípio menos esforço do que no caso sob demanda. Além disso, o método de sincronização deve ser simples e rápido para cada novo vídeo acrescentado ao conjunto de entrada.

Como o atraso entre os fluxos de vídeo é pequeno e limitado ao atraso da transmissão e decodificação, sendo ainda menor do que o tamanho do próprio vídeo, o uso da percepção humana para sincronizar esses fluxos torna-se mais simples. A tarefa de um *worker* consiste em definir os atrasos necessários em cada fluxo de forma que sua apresentação conjunta pareça sincronizada.

Num cenário em que o conjunto de entrada é formado por vídeos previamente armazenados e acessados sob demanda, a etapa de sincronização pode ocorrer após a captura dos vídeos. Diferente do caso ao vivo, no qual o espaço de busca da solução se restringe aos quadros de vídeo transmitidos no momento, no caso sob demanda, o ponto de sincronização buscado pode estar em qualquer parte do conteúdo do vídeo. Isso obriga que, no pior caso, todo o conteúdo de um vídeo tenha que ser processado para que se conclua que não existe nenhuma correlação entre o seu conteúdo e os dos outros vídeos do conjunto.

4.1.1.5. Elementos de interesse no conteúdo

A tarefa de encontrar pontos de sincronização, tanto com o uso da *crowd* quanto com o de soluções automáticas pode ser simplificada se existirem algumas ocorrências particulares nos conteúdos capturados. De acordo (STEINER, VERBORGH, *et al.*, 2011), estes instantes particulares definem mudanças

perceptíveis no conteúdo visual do vídeo causadas por (i) cortes e/ou transições de cenas; (ii) aparecimento e/ou desaparecimento de entidades específicas (como pessoas ou objetos) nas cenas e (iii) situações de particular interesse para o espectador (*highlights* (MARQUES NETO e SANTOS, 2010)). Mudanças na entonação dos personagens, na intensidade da narração e sons característicos (aplausos, explosões, silêncio, etc.) também podem ser utilizadas para identificar a ocorrência de situações particulares do conteúdo.

4.1.2. A *Crowd* nas etapas de sincronização

A Figura 12 ilustra o processo geral para sincronização de um conjunto de vídeos correlacionados a um evento. O conjunto de entrada é formado por *UGVs*, que são processados para que seus conteúdos sejam alinhados no tempo e apresentados de forma sincronizada.

Figura 12 - Processo de sincronização

Fonte: Elaborada pelo autor

O resultado da etapa de processamento é uma estrutura que armazena os pontos de sincronização entre cada par de vídeos com conteúdos correlacionados. A estrutura gerada pode ser usada para gerar diferentes apresentações combinando estes vídeos. Um mosaico de vídeos sincronizados sob a forma de uma matriz, a partir da qual um espectador pode selecionar um vídeo ou uma trilha de áudio de seu interesse é um exemplo desse tipo de apresentação. Outro exemplo foi apresentado por Arev *et al.* (2014). Os autores identificam ocorrências de interesse no conteúdo para produzir um vídeo coerente e "segmentado" de um evento. De fato, tais ocorrências definem pontos de sincronização que, em conjunto com diretrizes cinematográficas, são usados para selecionar quais câmeras devem ser cortadas e a duração destes cortes na produção do vídeo que reconta a história do evento.

A *crowd* pode desempenhar várias tarefas envolvidas do processo de sincronização. Estas tarefas vão desde a criação do *UGV* (*input*) até a geração de um valor de consenso para especificação de sincronização. Dentre estas tarefas, podem ser destacadas:

- Captura de vídeos, onde cada indivíduo grava seu próprio vídeo de um evento, gerando diferentes mídias e tornando-as disponíveis;
- Busca, anotação e o agrupamento de vídeos que pertencem a um mesmo evento;
- Especificação de pontos de sincronização, a partir de análise do conteúdo audiovisual dos vídeos;
- Verificação, ajustes e realização de consenso das especificações de sincronização entre os vídeos.

As atividades de captura, busca, anotação e agrupamento de vídeos estão fora do escopo da tese. Conforme as considerações apresentadas na seção 3.3, o uso de uma *crowd* para realizar a sincronização do conteúdo de vídeos ainda não havia sido reportado na literatura no momento da escrita desta tese, sendo a principal contribuição deste trabalho. Além desta atividade, que está no cerne do processo proposto, esta tese estuda o uso dos *workers* para executar atividades ligadas à aferição de qualidade dos resultados gerados em *crowdsourcing*. Os trabalhos de Schweiger *et al.* (2013) e Keimel *et al.* (2012) seguem a mesma direção, usando *crowds* para aferir a QoS de *streams* de vídeos providas pela Internet.

4.1.3. Gerenciamento das Contribuições

O gerenciamento das contribuições dos *workers* é um requisito herdado pelo uso de *crowdsourcing* no processo de sincronização. É necessário receber, distribuir e processar as contribuições da *crowd* para encontrar os *offsets* entre os *vídeos*.

Um dos princípios do *crowdsourcing* é a coleta das contribuições de cada *worker* e, com base nessas contribuições, encontrar a solução para um problema. No presente trabalho, a *crowd* assiste aos vídeos e encontra pontos de sincronização entre eles, se existirem. As contribuições enviadas pelos *workers* devem ser agrupadas com o objetivo de gerar um resultado final, que no caso deste trabalho, pode ser um dos valores de *offset* especificados para um par de vídeos e usado como para a sincronização dos seus conteúdos.

O gerenciamento das contribuições deve cuidar também da convergência dos valores das contribuições individuais de cada um dos *workers* para um valor que as

represente. Diversas funções podem ser utilizadas para determinar um valor médio, como será visto nos estudos de caso realizados.

Outra preocupação no gerenciamento de contribuições é a seleção dos pares de vídeos a serem enviados aos *workers*. Neste caso, deve-se descobrir qual par de vídeos deve ser enviado a cada momento para cada *worker*. Uma escolha adequada dos vídeos pode implicar em uma convergência mais rápida dos valores de *offset* gerados pelos *workers* e, como consequência, reduzir o número de contribuições necessárias para a solução do problema. Mais ainda, o método de seleção implementado deve garantir que os vídeos com poucas contribuições tenham prioridade de serem enviados aos *workers* a cada instante e que os vídeos para os quais os valores das contribuições já tenham convergido, não sejam mais selecionados.

4.1.4. Tarefas enviadas para a *crowd*

As tarefas são um aspecto essencial em processos *crowdsourcing*. O formato em geral das tarefas a serem executadas pelos *workers* para encontrar pontos de sincronização entre vídeos pode variar bastante. Algumas das propostas surgidas durante a pesquisa para formatar as tarefas enviadas à *crowd* são apresentadas a seguir.

4.1.4.1. Tarefa baseada em segmentos de vídeo

O sistema *reCAPTCHA* (VON AHN, MAURER, *et al.*, 2008) é apresentado como um sistema que usa recursos humanos para digitalizar material impresso que não possui versões digitais. No sistema, as palavras, que não puderam ser reconhecidas pelos métodos computacionais atuais, são enviadas aos usuários, que digitam as palavras lidas. Os usuários não digitalizam um livro inteiro, mas apenas uma palavra por vez. As inúmeras pequenas contribuições dos inúmeros usuários permitem a digitalização completa do livro. Assim, a solução de um problema que exigiria um grande esforço de um único usuário pode ser realizada com pequenas colaborações individuais provenientes da *crowd*.

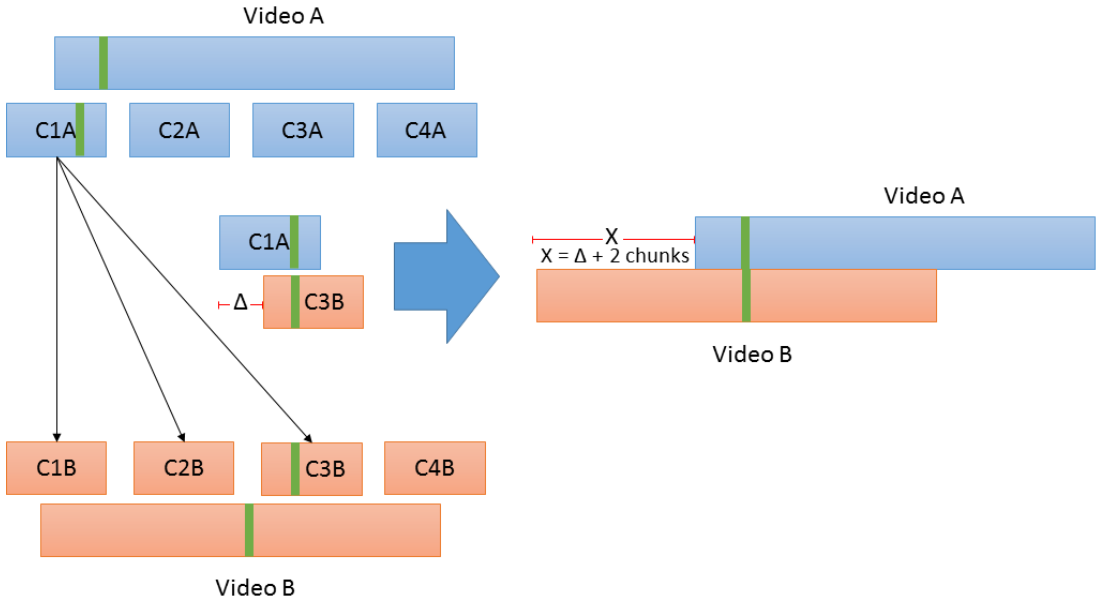
Uma abordagem semelhante pode ser adotada para o processamento de vídeos. Pode não ser prático pedir aos usuários que assistam todo o conteúdo dos vídeos para realizar uma tarefa, uma vez que a duração destes pode variar de

alguns segundos a horas. Desta forma, os vídeos a serem analisados precisam ser segmentados em intervalos de tempo (*chunks*), numa abordagem semelhante ao uso de palavras em vez de um texto inteiro no sistema *reCaptcha*. Um par de segmentos é então enviado a cada *worker* para que ele execute a tarefa de encontrar um ponto de sincronização entre os segmentos.

Assim, cada *worker* deve realizar uma tarefa que consiste em comparar se dois segmentos pertencentes a dois vídeos do conjunto de entrada possuem um ponto de sincronização, se o possuírem, o valor de Δ deve ser armazenado para que suas apresentações sejam sincronizadas.

A Figura 13 ilustra o resultado da execução das tarefas para permitir a sincronização de dois vídeos *A* e *B*. Primeiro, cada vídeo *A* e *B* é segmentado. Um par de segmentos é enviado para um *worker* que avaliará se há correlação entre os conteúdos dos segmentos e qual é o Δ entre eles, caso haja.

Figura 13 - Tarefa baseada em segmento



Fonte: Elaborada pelo autor

No exemplo da Figura 13, se os segmentos $[C_1A, C_1B]$ não puderem ser sincronizados, o próximo par é passado como tarefa, $[C_1A, C_2B]$, e assim por diante até ser encontrado o ponto de sincronização entre dois segmentos quaisquer ou até todos os pares terem sido distribuídos, indicando que não há sincronização entre A e B. No exemplo, o par $[C_1A, C_3B]$ contém um ponto de sincronização. O worker identifica e descobre o offset entre eles (Δ). Usando o valor de Δ e sabendo quais os segmentos continham o ponto de sincronização, é possível sincronizar todo o

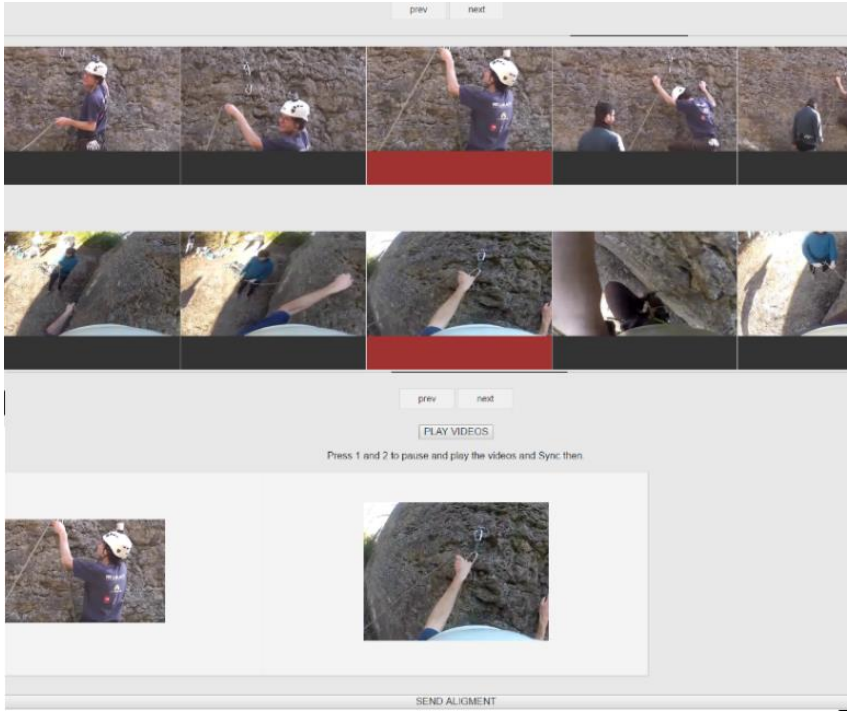
conteúdo dos vídeos [A, B]. Nesse caso, a diferença final entre os dois vídeos [A, B] é Δ mais duas vezes o tamanho do segmento, pois Δ foi encontrado na relação entre o primeiro pedaço de A e o terceiro de B, uma diferença de dois segmentos.

4.1.4.2. Tarefa baseada em sequências de keyframes

A sincronização baseada em *keyframes* pode ser vista como uma tentativa de melhoria da solução baseada em segmentos. Ao invés de trabalhar em cima de trechos curtos dos vídeos, cada *worker* recebe as sequências de *keyframes* dos pares de vídeos a serem sincronizados. Com o acesso facilitado ao conteúdo total dos vídeos de forma resumida, a possibilidade de encontrar um ponto de sincronização pelo *worker* é aumentada. No entanto, como o *worker* interage com *frames*, a precisão dos valores de Δ tende a ser mais baixa do que na abordagem baseada em segmentos. Uma forma de contornar o problema é dividir as tarefas em dois níveis de granularidade. No primeiro, baseado em *frames*, o *worker* procura por frames com similaridade visual, determinando um valor de Δ aproximado. No segundo, o *worker* assiste aos vídeos próximo à sincronização com base no Δ aproximado e tenta refinar esta informação, tornando a sincronização mais precisa.

A Figura 14 mostra um exemplo de aplicação da abordagem. No topo da interface, o *worker* pode selecionar os *keyframes* que acredita estarem perto do ponto de sincronização. Em seguida, os vídeos são reproduzidos a partir dos *frames* selecionados, permitindo ao *worker* identificar se esta seleção foi correta, se é necessário um refinamento ou se os *keyframes* foram incorretamente selecionados. Se a seleção de *keyframes* estiver correta, o *worker* confirma o alinhamento dos conteúdos. Do contrário, o *worker* tenta o ponto de sincronização indicado mais uma vez ou o descarta, por considerá-lo incorreto.

Figura 14 - Tarefa baseada em *keyframes*



Fonte: Elaborada pelo autor

4.1.4.3. Tarefa com vídeos completos

O uso de segmentos de vídeo visa reduzir o esforço dos *workers*, uma vez que eles analisam apenas pequenos trechos do vídeo a cada tarefa. Alguns dos problemas relacionados à abordagem de segmentos são (i) a exigência de um número muito grande de tarefas mesmo para vídeos relativamente curtos; (ii) a alta taxa de insucesso dos *workers* em encontrar pontos de sincronização entre segmentos e (iii) como consequência de (ii), os *workers* são desestimulados a continuar contribuindo com outros pares de segmentos.

A solução baseada na análise de *frames* se mostrou muito mais adequada à proposta desta tese. Como os *workers* assistem a “versões resumidas” do conteúdo para buscar de pontos de sincronização, o número de contribuições da *crowd* é sensivelmente reduzido, além de aumentar as chances de sucesso do *worker* na execução desta tarefa. Em contrapartida, os *frames* removem dois recursos importantes que poderiam auxiliar os *workers* na realização das tarefas: (i) informações de movimento da cena são perdidos e (ii) a análise do conteúdo é apenas visual, já que não é possível sumarizar as informações da trilha de áudio.

Como foi mencionado, a abordagem por *keyframes* obriga que o processo de sincronização inclua uma etapa de refinamento dos valores de atraso entre os pares

de vídeo definidos pelos *workers*. Isso adiciona um esforço extra para a realização da tarefa pelo *worker*. Uma forma de compensar o problema é utilizar o cascateamento de tarefas (a finalização de uma tarefa habilita a próxima) como forma de realizar processamentos mais complexos, como proposto por Amorim, Santos, Mendes & Tavares (2017). A necessidade de um pré-processamento dos vídeos do conjunto de entrada a fim de realizar a extração dos *frames* chave é outra desvantagem desta última abordagem.

Dessa forma o uso de vídeos completos é uma alternativa que pode ser utilizada para a sincronização dos vídeos. Como forma de amenizar o problema dos *workers* assistirem todo o vídeo para realizar a análise, ferramentas devem ser adicionadas aos players, como a possibilidade de avançar no tempo e permitir a navegação pela barra de tempo do vídeo, fazendo com que o *worker* possa manipular os vídeos em busca do ponto de sincronização.

4.1.4.4. Tarefa para sincronização ao vivo

A tarefa de sincronizar fluxos de vídeo ao vivo não requer a análise de todo o conteúdo dos vídeos envolvidos a fim de torná-los sincronizados. Isso ocorre porque o evento está sendo transmitindo ao mesmo tempo em que os vídeos são capturados. Assim, a falta de sincronização durante a reprodução combinada desses vídeos é causada por atrasos e *jitter* na transmissão de cada um dos fluxos aos usuários. Mais ainda, a tarefa de determinar os atrasos (valores de Δ) entre os fluxos para que eles se sincronizem consiste em (i) fixar um vídeo do grupo como referência (em geral, o mais atrasado) e (ii) ir ajustando os valores de atraso a cada novo fluxo adicionado ao grupo de vídeos a ser combinado.

Desta forma, a sincronização pela *crowd* pode ser alcançada usando ferramentas que permitem ao usuário manipular o tempo de apresentação dos vídeos. Tipicamente, o *worker* deve pausar os vídeos do conjunto alternadamente, de forma a inserir atrasos inserindo atrasos sucessivos até atingir o ponto ideal de sincronização entre eles. Essa sincronização pode ser repassada aos demais usuários que acessem o par, permitindo a sincronização imediata dessa nova apresentação, sem a necessidade de uma nova sincronização neste novo cliente.

4.2. CONSIDERAÇÕES

Neste capítulo foram discutidos diversos aspectos a serem considerados quando uma *crowd* é recrutada para solucionar o problema de sincronização de um conjunto de vídeos com conteúdo correlacionado. Diferentes tipos de tarefas propostas e utilizadas durante a pesquisa foram analisados, permitindo que fossem estabelecidos uma série de critérios relacionados a construção destas tarefas, tais como a sua duração, tipo, quantidade e convergência dos resultados alcançados, entre outros critérios.

Alguns aspectos, porém, foram desconsiderados no processo de sincronização mais amplo, e deverão ser abordados como etapas futuras da pesquisa. Dentre eles, devem ser destacadas as que envolvem: (i) a captura, busca, anotação e agrupamento do conjunto de vídeos de entrada; (ii) a otimização da execução das tarefas em termos de tempo, custo, número de *workers*; (iii) o *workflow* para tarefas complexas; (iv) o recrutamento dos workers e o controle dos resultados produzidos e ainda, (v) a interação dos *workers* com as ferramentas necessárias para execução das tarefas e a qualidade dos seus resultados.

O próximo capítulo discute os diversos aspectos comuns às abordagens baseadas em *crowdsourcing* com o objetivo de propor um modelo funcional para auxiliar o desenvolvimento de sistemas que utilizem a *crowd* para realizar algum tipo de processamento sobre vídeos.

5. MODELO *CROWDVIDEO*

Nos capítulos anteriores, foram apresentados os conceitos de *crowdsourcing*, um estudo sobre a sua aplicação para a solução de problemas que envolvem a execução de tarefas centradas no conteúdo de vídeo, e um estudo específico do uso de *crowdsourcing* na sincronização de vídeos.

Um dos objetivos destes estudos foi explicitar as principais características das aplicações de *crowdsourcing* que manipulam o conteúdo dos vídeos para solucionar problemas e categorizar os trabalhos a partir das características encontradas, em especial no contexto de sincronização de vídeos. Fora a sincronização descrita no capítulo 4, os *workers* geralmente realizam as seguintes atividades envolvendo vídeos, como mostrado com a apresentação dos estudos primários do mapeamento no capítulo 3:

- Anotar vídeos, que corresponde a edição do conteúdo, adicionando novas informações (descrição do conteúdo, legendas, rankings, ...) ao vídeo. A *crowd* pode ser usada para classificar objetos usando ferramentas gráficas, criar resumos e *How-to-Videos*, encontrar cenas-chave em vídeos do YouTube ou até mesmo atuar na vigilância, onde a *crowd* assiste às filmagens procurando por ações ou indivíduos suspeitos nas cenas;
- Identificar pontos-chave no conteúdo dos vídeos: esses pontos são usados para encontrar o vídeo em uma base, ou gerar um resumo do vídeo. Esse processo pode ser realizado por contribuições diretas ou indiretas. Comentários e reações do Twitter para encontrar os melhores momentos em uma partida de futebol são exemplos de contribuições indiretas e a definição de mudanças de cena, de contribuições diretas;
- Recomendar e recuperar: recursos de busca e recomendação são baseados no resultado de uma análise de conteúdo do vídeo pela crowd. A *crowd* fornece um conjunto de dados indexáveis para dar suporte ao processo de busca e acesso ao conteúdo de interesse de outros usuários;
- Avaliar e aferir qualidade: a percepção dos *workers* é usada para avaliar a qualidade audiovisual dos vídeos, que pode variar de acordo com as

características do codificador, dispositivo de apresentação, conectividade, entre outros fatores do contexto de apresentação.

Outra atividade comum atribuída à *crowd* é a geração de vídeos. Esta atividade, porém, se diferencia das demais por não se tratar de uma atividade de *Distributed Human Intelligence Tasking*, e não será considerada para o contexto deste capítulo. As demais atividades se caracterizam por serem dependentes da percepção humana e não são facilmente descritas por meio de algoritmos executáveis por máquinas. Alguns trabalhos utilizam o resultado gerado pela *crowd* como forma de treinar e validar soluções automáticas (Su (2012) por exemplo). Entretanto, a maioria dos trabalhos envolve o uso de *crowdsourcing* para anotar o conteúdo e avaliar a qualidade de vídeos.

Após desenvolver várias ferramentas para avaliar o uso da *crowd* na sincronização de vídeo, percebeu-se que elas compartilhavam vários aspectos em comum e que estes aspectos eram bastante semelhantes aos resultados do mapeamento sistemático reportados na seção 3.3.

5.1. CROWDVIDEO

A compreensão de como o conceito de *crowdsourcing* tem sido usado no processamento de vídeos e os estudos detalhados para criação de ferramentas, plataformas e técnicas para permitir a sincronização de vídeos apoiados neste conceito permitiram estabelecer uma base de requisitos para modelar as necessidades dos sistemas voltados ao processamento de vídeos usando a *crowd*. Dentre os principais recursos utilizados na construção de uma aplicação podem ser destacados: (i) distribuição das tarefas entre os usuários; (ii) armazenamento dos dados gerados; (iii) agregação das contribuições resultantes das tarefas de forma a gerar resultados; (iv) identificação de usuários mal-intencionados (*spammers*); e (v) gerenciamento de *workers* e da execução das tarefas.

O resultado deste processo de aprendizagem foi a elaboração de um modelo funcional, denominado *CrowdVideo*, para auxiliar a construção de aplicações que aplicam *crowdsourcing* no processamento de vídeos. O *CrowdVideo* possui como principais objetivos:

- Guiar a modelagem de aplicações que utilizem técnicas de *crowdsourcing* na manipulação de conteúdo de vídeos;
- Apresentar componentes customizáveis que possam ser implementados de acordo com a necessidade específica de cada aplicação;
- Definir modelos de agregação de dados para geração dos resultados esperados;
- Definir formas de distribuição de tarefas entre os membros da *crowd*;

O modelo inclui apenas os principais conceitos e relações a serem usados no desenvolvimento, ocultando detalhes internos. Ele pode ser utilizado por diferentes esquemas em diferentes aplicações. No âmbito deste trabalho, o *CrowdVideo* tem como objetivo apoiar o desenvolvimento de aplicações *crowdsourcing* baseadas em conteúdo de vídeo.

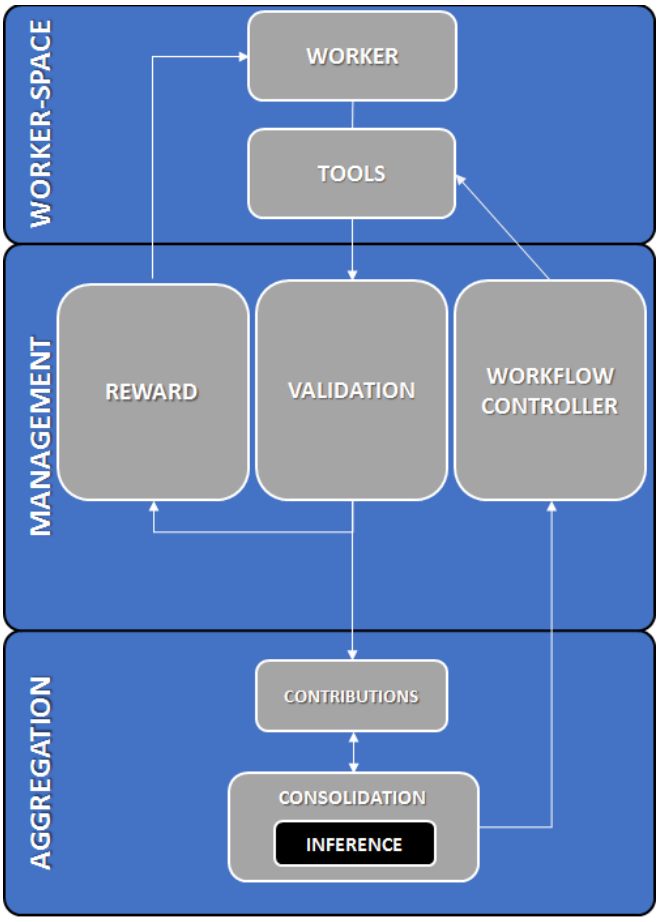
Para o *CrowdVideo*, toda interação (exceto as ligadas à reprodução do conteúdo) da *crowd* sobre um vídeo é considerada como uma anotação. Anotar é o ato de adicionar dados a outro dado, estabelecendo dentro de um contexto a relação entre o dado anotado e o dado de anotação (OREN, 2006).

Neste trabalho, o dado a ser anotado constitui o objeto de vídeo, e a anotação o resultado do trabalho de cada *worker* usando um determinado recurso (por exemplo, uma interface Web com um player de vídeo). Por exemplo, um vídeo que tem sua qualidade testada em uma escala, pode ser interpretado como um vídeo sendo anotado, onde a escala escolhida pelo usuário é uma anotação relacionada ao vídeo. Semelhante ocorre na sincronização de vídeos, onde a análise dos vídeos gera uma informação temporal sobre os mesmos, o que é descrito no capítulo 2 como acopladores.

5.2. COMPONENTES DO MODELO FUNCIONAL

O modelo funcional *CrowdVideo* foi dividido conceitualmente nos três componentes ilustrados na Figura 15: *WORKER-SPACE LEVEL*, *MANAGEMENT LEVEL* e *AGGREGATION LEVEL*.

Figura 15 - Modelo *CrowdVideo*



Fonte: Elaborada pelo autor

O primeiro nível inclui os elementos de comunicação com a *crowd*. É nele que o *worker* recebe suas *tasks* (atividade a ser realizada por um *worker*), gera e envia suas contribuições. O segundo nível cuida das atividades relacionadas à gestão das *tasks*, como distribuição das tarefas, validação das contribuições e gerenciamento das recompensas. Por fim, o terceiro nível é responsável pela consolidação das contribuições. Neste nível, são identificadas quais contribuições resultaram em uma resposta definitiva a uma tarefa, quais tarefas estão tendo problemas para serem resolvidas e determina quando um resultado é alcançado.

5.3. Worker-Space

O primeiro macro componente modela os insumos necessários para a realização da tarefa pelo *worker* e para a coleta das contribuições produzidos por ele. O *worker* deverá utilizar uma ferramenta para dar suporte à execução da sua tarefa sobre um objeto de vídeo. O objeto, a plataforma e a ferramenta definem o *worker-space* para a realização da tarefa. A ferramenta é a interface entre a *crowd* e

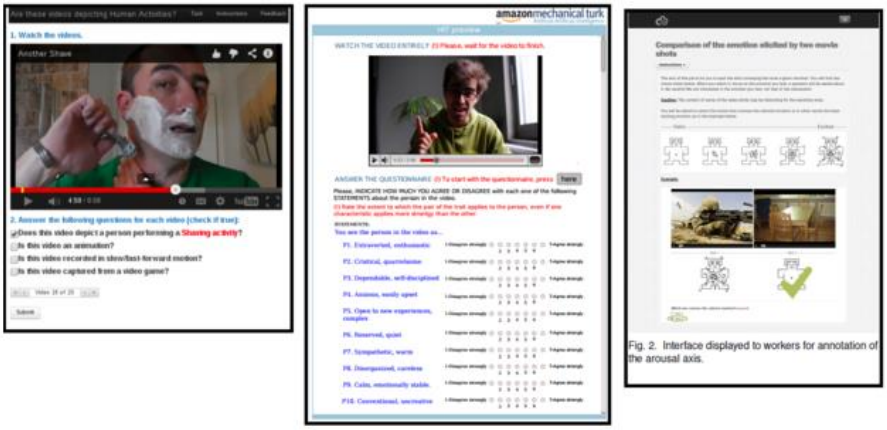
a *task*. Ela deve conter todas as funcionalidades necessárias para que cada *worker* possa realizar sua tarefa.

Sendo assim, é comum que cada tarefa exija uma interface customizada para os diferentes tipos de atividades e abordagens apresentadas aos *workers*. No entanto alguns elementos são comuns a muitos problemas, e problemas com um contexto próximo possuem aspectos semelhantes em suas ferramentas e interfaces. Um elemento comum entre aplicações de vídeo e *crowdsourcing* é a presença de um *player* como elemento central. Esse *player* irá apresentar o objeto de vídeo base para a tarefa. Em torno do vídeo são adicionados elementos específicos a cada tarefa, como campos para a entrada de anotações, escalas de qualidade, e perguntas sobre emoções e ações.

A Figura 16 mostra o padrão para interfaces de avaliação de qualidade de vídeos: o player de vídeo em posição central, com instruções sobre a tarefa a ser realizada próximo à janela do player e as questões de avaliação logo em seguida. Na terceira interface, à direita, ocorre uma pequena variação: são apresentados dois vídeos ao mesmo tempo e a avaliação é feita sobre a relação entre ambos, por exemplo, qual demonstra melhor uma determinada emoção.

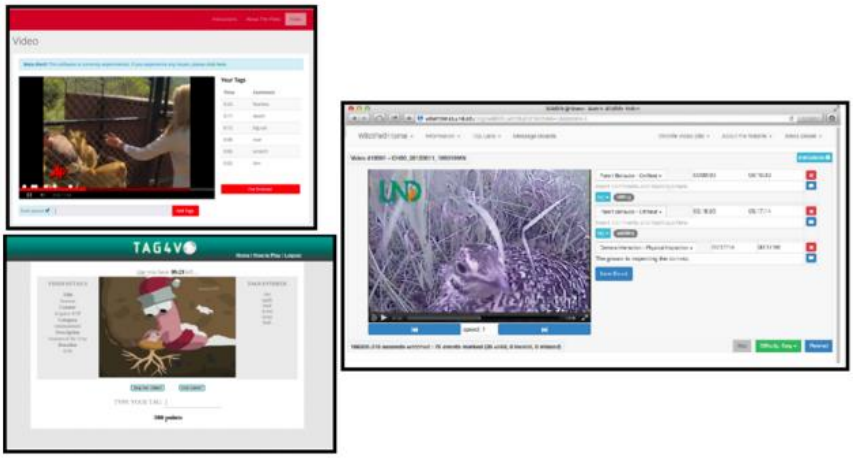
A Figura 17 apresenta exemplos de interfaces para anotação. Novamente o vídeo ocorre como objeto principal da interface, porém cada aplicação tem sua própria forma de anotá-los. À esquerda superior, os *workers* adicionam rótulos relacionados a cada momento específico do vídeo, anotando em um dado tempo exato do vídeo; à esquerda inferior, cada anotação faz parte do vídeo completo, não importando para qual tempo do vídeo foi gerado o rótulo. À direita, a *crowd* deve selecionar momentos específicos do vídeo nos quais os pássaros aparecem e dizer o que eles fazem.

Figura 16 - Exemplos de interfaces de avaliação de vídeos



Fontes: (HEILBRON e NIEBLES, 2014) (esquerda), (ARAN, BIEL e GATICA-PEREZ, 2014) (centro) e (BAVEYE, DELLANDREA, *et al.*, 2015) (direita)

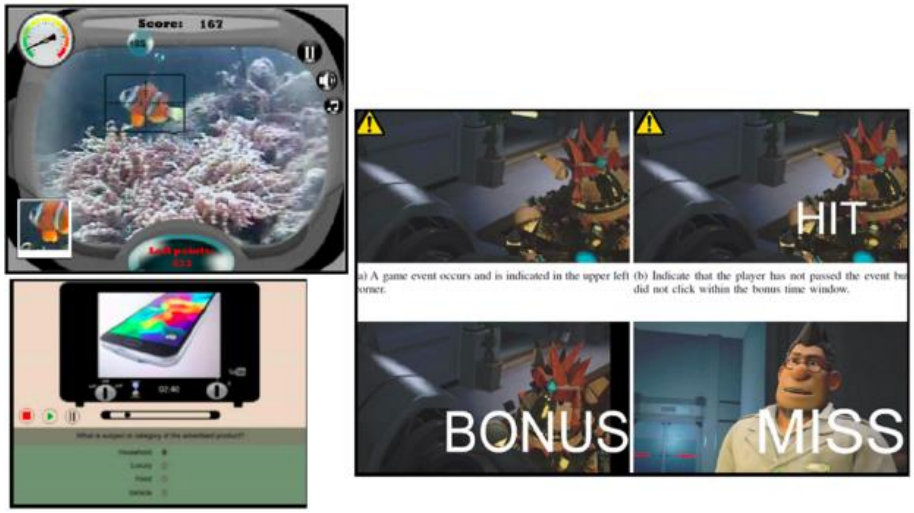
Figura 17 - Exemplo de interfaces de anotação de vídeos



Fonte: integração de (CRAGGS, KILGALLON SCOTT e ALEXANDER, 2014) (esquerda superior), (PINTO e VIANA, 2013) (esquerda inferior) e (DESELL, GOEHNER, *et al.*, 2015)(direita)

A Figura 18 apresenta interfaces que fazem uso de elementos de jogos para melhorar o desempenho e participação da *crowd* em relação à tarefa. À esquerda superior, o jogo consiste em usar um alvo que se move junto ao cursor do mouse para capturar os peixes que aparecem no vídeo, anotando as posições dos peixes no vídeo; à esquerda inferior é apresentado um jogo no estilo *quiz*, onde à medida que o vídeo passa é perguntado que tipo de objeto aparece na tela; por fim, à direita há um jogo para testar a sincronia IDMS entre vídeos, onde os membros da *crowd* devem reagir a ações durante a execução vídeo. O tempo dessa ação é utilizado para avaliar se a sincronia foi atingida, e o *worker* recebe pontos por sua reação.

Figura 18 - Outros exemplos de interface



Fonte: integração de (DI SALVO, GIORDANO e KAVASIDIS, 2013) (esquerda) e (RAINER, PETSCHARNIG, *et al.*, 2015) (direita)

Outro elemento comum às interfaces de *crowdsourcing*, não apenas de aplicações em vídeo, está relacionado à recompensa dada ao *worker*. É comum a interface apresentar a pontuação, ranking ou remuneração ganha, buscando estimular a participação do *worker* e seu empenho na realização com sucesso das tarefas.

5.4.Management

O gerenciamento trata das questões: (i) de controle do fluxo de execução e distribuição de tarefas (*workflow*); (ii) da validação das contribuições a cada etapa de execução; e (iii) da recompensa aos *workers*.

5.4.1. Workflow Controller

O controlador de *worklow* (SU, 2012) (YANG, 2016) (USTALOV, 2015) é um componente que tem como principais funcionalidades: (i) controlar o fluxo de execução das tarefas e das contribuições produzidas e (ii) distribuir cada tarefa habilitada para cada um dos *workers* aptos a realizá-la. Uma tarefa está desabilitada se as condições necessárias para que um *worker* inicie sua execução (em geral, em termos de ordem, tempo e insumos utilizados) não estiverem sendo satisfeitas.

No modelo proposto, o *workflow controller* deve garantir que todos os vídeos recebam um número suficiente de contribuições válidas para a conclusão de uma

tarefa. Se a distribuição de tarefas for otimizada, pode-se esperar uma redução no número de contribuições necessárias.

Algumas formas identificadas de fazer esta distribuição são: de forma aleatória; fazendo todos *workers* acessarem todos os vídeos; deixando cada *worker* escolher qual tarefa deseja realizar.

Usar a opção de distribuir todos os vídeos para todos os membros depende de alguns fatores: *crowd* e quantidade de vídeos limitados, e um número de contribuições ilimitada. O segundo fator está relacionado aos recursos para a solução. Como todos os *workers* devem interagir com todos vídeos, o número de contribuições será elevado e será necessário ter recursos suficientes para cobrir as contribuições. Esse esquema é ideal quando há um grupo limitado de membros na *crowd* e deseja-se uma cobertura total de todos os vídeos com o máximo de contribuições.

Outra forma de distribuição é aquela na qual *workers* escolhem quais tarefas preferem realizar, para as quais eles estejam aptos. Isto garante um melhor engajamento pelo *worker*, já que ele está escolhendo sua tarefa. Tarefas que não são escolhidas deverão ser enviadas a membros da *crowd* sem preferências, ou imposta em um certo ponto.

A distribuição aleatória dos vídeos para os *workers* é a mais simples de ser implementada, porém pelo seu fator aleatório, existe uma dificuldade em direcionar as contribuições e coletar os resultados.

Porém, um fator não considerado nestas formas de distribuição é a realização do processamento dos dados em tempo real, já que na maioria dos casos estudados os dados são coletados para posterior processamento (*batch processing*). Tempo real refere-se ao fato de analisar as contribuições à medida que as mesmas são submetidas, e com isso, influenciando na distribuição das próximas tarefas. No processamento em tempo real, é necessário adicionar um novo fator às soluções: o estado atual da resposta, ou seja, considerando as contribuições que já foram realizadas e ainda quanto da resposta foi encontrada. O estado atual permite determinar se uma resposta deve ser considerada como correta, ou se mais contribuições devem ser requisitadas até que a resposta seja encontrada. Já o processamento *offline* depende apenas de quantas contribuições devem ser

coletadas para que a distribuição seja encerrada, sem essa análise constante do estado da solução.

Uma forma proposta de realizar a distribuição de tarefas *online* consiste em priorizar as tarefas de forma a convergir o maior número de tarefas, e a partir delas inferir as demais, reduzindo a quantidade de contribuições necessárias. Esta abordagem possui as seguintes propriedades:

- Cada possível tarefa possui um número mínimo *n* de contribuições necessárias para convergir (ser considerada resolvida) > 0;
- Uma tarefa converge quando se tem um grau de certeza (*α*) de que uma resposta àquela tarefa foi encontrada, não sendo necessário repetir a tarefa por outro membro;
- O componente de distribuição interage diretamente com o algoritmo de convergência;
- Os membros da *crowd* podem ser desconhecidos e não identificados;

A distribuição proposta funciona seguindo o algoritmo:

```
DISTRIBUTION (TASKS)
1.  L = INIT_LIST (TASKS, PARAM);
2.  L_AUX = NULL;
3.  WHILE (L != NULL)
4.      TASK = REMOVE_TASK(L);
5.      IF TASK.NOTCONVERGED
6.          L_AUX.INSERE(TASK);
7.      IF L_AUX != NULL
8.          L_AUX.ORDERBYALPHA()
9.      L = L_AUX
10.     GOTO 2
```

O algoritmo de distribuição distribui os vídeos (ou parte deles) e as tarefas aos *workers*. Os vídeos funcionam como insumos para que os *workers* realizem suas tarefas. Primeiramente, os vídeos são relacionados às tarefas de acordo com o *workflow* de cada aplicação. Estas tarefas são armazenadas em uma lista (***L***).

A ordem de inicialização das tarefas na lista ***L*** pode variar de acordo com o problema. Inicialmente podem ser alocadas com o critério de duração para que as tarefas com vídeos mais longos sejam anotados primeiro, ou talvez os mais curtos para gerar mais contribuições em vídeos diferentes em menor tempo, sendo então

um parâmetro para a função de inicialização da lista. Junto à lista ***L***, é criada uma lista auxiliar (***L_AUX***) vazia, que irá servir para armazenar as tarefas que já tiveram contribuições na rodada, mas que ainda não convergiram.

Cada tarefa na lista principal ***L*** é enviada a um *worker*, e então removida da lista. Caso as contribuições geradas pelo *worker* façam aquela tarefa convergir, ela não ficará mais em nenhuma lista, pois não há mais necessidade de contribuições, caso contrário, ela é inserida na lista auxiliar de forma a aguardar a próxima rodada de contribuições. Caso uma tarefa enviada para um *worker* não seja executada em tempo hábil, a tarefa é restaurada ao estado anterior ao envio, retornado à lista ***L*** para que seja novamente executada por um *worker*.

Uma vez que não há mais tarefas na lista principal, a lista auxiliar é ordenada de acordo com o grau de convergência atual. Aqueles que possuem um grau de convergência maior são colocados à frente na lista, para que possam mais rapidamente convergir. Caso essa lista auxiliar esteja vazia, significa que todas as tarefas convergiram e não há mais nada o que fazer, o problema foi solucionado. Caso contrário, a lista ***L*** recebe ***L_AUX*** e o processo recomeça. Como forma de otimização, a inserção dos elementos na lista auxiliar pode ser feita de forma ordenada, sendo desnecessário a ordenação na linha 8.

5.4.2. Validação

As aplicações necessitam tratar o problema de contribuições maliciosas, ou seja, contribuições geradas pelos chamados *spammers*. Nesse caso, o membro não tenta realizar a tarefa, ele apenas quer terminá-la e receber a recompensa prometida ou no pior caso comprometer as atividades envolvidas. Desta forma, estas contribuições devem ser encontradas e tratadas, a fim de não contaminar os resultados.

A técnica mais utilizada para esse tratamento é o uso de *Gold Standard*s (*Ground Truth/ Control Questions*). Essa técnica se baseia no fato de que se um *worker* erra muitas questões, é possível que ele seja um *spammer* (LE, EDMONDS, *et al.*, 2010). O método para tratar o problema consiste em adicionar uma etapa a mais na tarefa a ser realizada pelo *worker*, para a qual a resposta é conhecida. Se o *worker* errar esta etapa cujo resultado é conhecido, todas as demais contribuições

dele deverão ser desconsideradas, pois ele passa a ser classificado como um *worker* não confiável.

Nas aplicações que envolvem tarefas sobre o conteúdo de vídeos, ao invés de utilizar técnicas convencionais (como questões de controle), podem ser utilizadas técnicas baseadas em características específicas do objeto vídeo. Um exemplo é analisar quanto tempo um usuário leva para responder uma pergunta relacionada ao vídeo (RAINER e TIMMERER, 2014) (ANEGEKUH, SUN e IFEACHOR, 2014). Isto permite saber se o *worker* ao menos assistiu ao vídeo antes de realizar a contribuição. Caso esse tempo seja menor que o tempo de duração do vídeo, a contribuição pode ser considerada falha.

5.4.3. Recompensa

Os sistemas de recompensa servem para encorajar os membros da *crowd* a participar na execução de tarefas e fazer com que os mesmos retornem para fazer novas tarefas, uma vez que sem *workers*, não há contribuições.

A recompensa recebida por um *worker* pode ser atrelada ou não ao seu desempenho. Em muitas plataformas de *crowdsourcing* (SCEKIC, 2013), a recompensa em dinheiro é a mais comum. Em casos relacionados a soluções cooperativas de problemas com contribuições voluntárias explícitas (SCEKIC, 2013), os *workers* recebem recompensas em termos de reputação, satisfação pessoal, diversão, entre outras modalidades. Para o uso de pagamento, existem APIs, como a API das plataformas *Crowdflower* e *Amazon Mechanical Turk*. Além de permitir gerenciar o pagamento das tarefas, tais plataformas possuem uma série de usuários cadastrados que podem atuar como *workers* da aplicação.

Contribuições voluntárias implícitas normalmente não estão associadas à realização de tarefas pelos *workers*, mas ao fornecimento de informações (por meio de *piggybacking*) para a solução de um problema global. Este tipo de contribuição não caracteriza problemas do tipo *Distributed Human Intelligence Tasking* e, portanto, estão fora do escopo deste modelo.

A utilização de membros de equipes de desenvolvimento e pesquisa como parte de uma *crowd* é uma prática comum no meio acadêmico. Pesquisadores por muitas vezes fazem com que estudantes participem dos experimentos de forma a

receberem recompensas não econômicas, mas sim acadêmicas tais como notas e créditos extra, criando muitas vezes *crowds* especialistas para a resolução das tarefas.

Outra forma comum de recrutar contribuidores é por meio de uma chamada aberta à participação de voluntários. Quando voluntários realizam as tarefas do sistema, uma das melhores práticas (HOßFELD, KEIMEL, *et al.*, 2014) consiste em utilizar componentes de jogos (gamificação) nas aplicações. O uso de gamificação pode fazer com que até 80% dos contribuidores retornem à aplicação e realizem novas contribuições, enquanto que sistemas regulares de tarefas retém apenas cerca de 23% da *crowd* (HOßFELD, KEIMEL, *et al.*, 2014). Além de uma taxa elevada de contribuidores que retornam à aplicação, quando jogos são envolvidos na tarefa, o número de contribuições não confiáveis pode cair de 13,5% para 2,3%.

Ainda relacionado às recompensas, a aplicação deve considerar se irá possuir usuários anônimos, ou se manterá referências de quais membros da *crowd* realizaram contribuições. Essa decisão impacta em dois fatores: nas recompensas e nos métodos de convergência de dados. No caso de contribuidores voluntários, caberá ao *crowdsourcer* criar atrativos na aplicação para que obtenha o número suficiente de contribuições para resolver seu problema.

5.5. Agregação

A agregação faz a análise dos dados gerados pelas contribuições dos *workers*. O objetivo desta análise é agrupar as contribuições geradas pelos *workers*, a fim de gerar a solução para o problema a ser resolvido pela aplicação. O *workflow* utilizado para descrever o fluxo de execução de uma aplicação, pode estabelecer diferentes níveis de tarefas e também dependências entre elas e os resultados produzidos a cada nível de execução. Neste caso a agregação pode ser utilizada para informar aos diferentes níveis, quais tarefas estão habilitadas e podem ser distribuídas.

5.5.1. Consolidação

A consolidação diz respeito a como juntar as contribuições realizadas pelos *workers* sobre uma tarefa com o objetivo de verificar se aquela tarefa chegou a um resultado ou não (convergiu ou não). Tarefas que convergiram no processo de consolidação, podem fazer parte direta da solução da aplicação, ou podem dar início

a uma nova fase de tarefas da aplicação, que por sua vez terão também que passar por todas as etapas, inclusive a consolidação das contribuições.

Tabela 1 - Fatores de decisão na escolha da técnica de agregação

Algorithm	Aggregation Model	Computing Model
MD	NON-ITERATIVE	ONLINE
HP	NON-ITERATIVE	ONLINE
ELICE	NON-ITERATIVE	ONLINE
EM	ITERATIVE	OFFLINE
SLME	ITERATIVE	OFFLINE
GLAD	ITERATIVE	OFFLINE
ITER	ITERATIVE	OFFLINE

Fonte: adaptado de (HUNG e AL., 2013)

A Tabela 1 apresenta métodos de consolidação (ou agregação) que podem funcionar de forma *online* ou *offline*, ou seja, o processamento das contribuições pode ser feito em tempo de execução, onde a cada nova contribuição a convergência é calculada a fim de saber quais tarefas já foram realizadas e se o processamento chegou ao fim, ou o processamento pode ser feito apenas após todas as contribuições serem realizadas.

O processamento em tempo real permite a otimização da quantidade de contribuições a serem realizadas, já que a distribuição só encerrará se todas tarefas convergirem. Porém, como esse processamento ocorre em tempo real, é necessário o uso de técnicas que possam ser executadas em tempo real. Segundo Huang (2013), dentre as técnicas de agregação mais conhecidas, as que podem ser utilizadas em tempo real são: *Majority Decision* (*MD*), *HoneyPot* (*HP*) e *Expert Label Injected Crowd Estimation* (*ELICE*).

O processamento *offline* permite aos algoritmos de agregação utilizar técnicas chamadas incrementais, onde a cada interação do algoritmo ele: atualiza o valor agregado de cada questão baseado na experiência de cada membro da *crowd* que respondeu aquela questão e em seguida ajusta a experiência de cada *worker* baseado nas respostas dadas por ele. Estes algoritmos exigem um elevado tempo de processamento, tornando-os inviáveis na abordagem *online*, mas uma excelente escolha para casos *offline*. São exemplos de algoritmos incrementais (HUNG e AL., 2013): *Expectation Maximization* (*EM*), *Supervised Learning from Multiple Experts* (*SLME*), *Generative Model of Labels, Abilities, and Difficulties* (*GLAD*) e *Iterative Learning* (*ITER*).

A escolha de uma destas técnicas depende ainda se a aplicação irá utilizar *workers* anônimos ou não. Isto ocorre pelo fato de apenas as técnicas de *MD* e *HP* poderem trabalhar com usuários anônimos, sem levar em consideração sua experiência na realização de tarefas. Todas as demais necessitam deste tipo de informação.

Um último fator determinante para a escolha de qual técnica de agregação utilizar, vem diretamente da dimensão do problema abordado: a quantidade possível de valores de saída da tarefa. Apenas *MD*, *HP* e *EM* podem ser utilizados quando mais de dois valores de saída são possíveis, enquanto que as demais citadas podem trabalhar apenas com dois possíveis valores de saída (0 ou 1, verdadeiro ou falso, ...) (HUNG e AL., 2013).

No entanto, diversas vezes a *crowd* é utilizada para realizar tarefas cuja solução não é constituída apenas de duas opções. Por diversas vezes uma tarefa pode ter ***n*** diferentes resultados. Por exemplo, no problema de avaliação de qualidade, é comum o uso da escala *MOS* (RIBEIRO, 2011) com várias opções.

No problema de anotação de vídeos, é possível que o *worker* anote qualquer instante do conteúdo do vídeo. Assim, ele pode gerar ***n*** anotações em diferentes pontos do vídeo, com ***n*** igual ao número de frames do mesmo. De fato, vários trabalhos que marcam/associam objetos ao conteúdo dos frames podem gerar outras ***m*** anotações por frame. Ou seja, é possível falar em ***n*m*** anotações sobre o conteúdo do vídeo.

Contudo, pode-se afirmar que as ***n*** possibilidades de anotação seguem uma distribuição categórica (AGRESTI, 2011). A distribuição categórica descreve a possibilidade dos resultados de um evento aleatório que pode ter *k* possíveis resultados, com a probabilidade de cada resultado especificado separadamente. Os ***k*** possíveis resultados são as categorias às quais as contribuições da *crowd* são mapeadas. Essas categorias podem ser representadas por intervalos de tempo nos vídeos, por exemplo, anotações feitas sobre um vídeo e que estejam em um intervalo de tempo menor que 1s (ou +-30 frames) se referem à mesma anotação. Assim, elas poderiam ser processadas como um conjunto e intervalos que não possuem nenhuma anotação poderiam ser desconsiderados. Não apenas os problemas contínuos podem ser considerados desta forma, mas problemas como o de qualidade que possuem *x* possibilidades de valores de qualidade podem ser

mapeados para *x* categorias, generalizando o problema para a distribuição categórica.

É possível descrever então que a função de anotação F(O,W,T) possui uma distribuição categórica cujo espaço amostral é um conjunto de ***k*** itens unicamente identificados que correspondem aos intervalos de anotação dos vídeos, descritos anteriormente, onde em cada intervalo estão ***x*** anotações. O espaço amostral é uma sequência finita de inteiros que servem de rótulo para cada categoria. Por exemplo, um espaço amostral {1, 2, ... , k} com ***k*** diferentes categorias. A função que descreve a probabilidade de que uma nova anotação seja alocada em uma das ***k*** categorias existentes é dada por:

$$f(x = i \mid \boldsymbol{p}) = p_i,$$

Onde $\boldsymbol{p} = (p_1, \ldots, p_k)$, p_i representa a probabilidade de encontrar o elemento ***i*** e $\sum_{i=1}^{k} p_i = 1$.

Estas categorias e probabilidades podem ser utilizadas para calcular a convergência das tarefas, descobrindo se as mesmas estão chegando a um valor comum ou se os valores estão dispersando. Sendo assim, dois limites de convergência são definidos: o superior que define o valor de probabilidade ***p*** que um grupo ***i*** deve atingir para ser o ponto de convergência e um valor inferior, que indicará uma dispersão das contribuições. Caso todas probabilidades $p_=(p_1,..., p_k)$, p_i sejam inferiores a esse limite inferior, ocorre uma divergência. Uma convergência ou divergência implica que não serão mais pedidas contribuições para aquela tarefa.

O uso da distribuição categórica, além de permitir a decisão de que uma tarefa convergiu ou não, traz um efeito colateral positivo interessante: a aplicação da distribuição categórica permite a eliminação de contribuições destoantes das demais ao encontrar o grupo de convergência. Com isso é possível identificar contribuições com altas taxas de erro, auxiliando na etapa de agregação e identificação de contribuições maliciosas.

A convergência pode ainda ser especializada para cada grupo de problemas. Por exemplo, quando se necessita adicionar *tags* aos vídeos, além dos pontos onde as *tags* são adicionadas, é importante verificar se essas *tags* que estão agrupadas se assemelham, ou seja, a convergência neste caso pode não considerar em que

tempo do vídeo as *tags* são colocadas, mas se as *tags* colocadas são semelhantes o suficiente para serem consideradas corretas.

Outra forma de verificar se tarefas convergiram, ou seja, se uma tarefa chegou a uma solução, é através de subtarefas. Para isso, é verificado se as respostas atribuídas a uma tarefa estão corretas através de subtarefas, onde o *worker* analisa se a tasks foram corretamente executadas, ou se é necessário gerar outra contribuição para tentar resolver aquela tarefa.

5.5.2. Inferência

A inferência é um subcomponente da consolidação. Ele consiste em descobrir o resultado de uma tarefa a partir do resultado de outras tarefas que já tenham convergido. Quando é possível identificar uma função de inferência entre as tarefas realizadas, há um ganho na finalização das tarefas. A inferência é realizada sobre resultados das contribuições dos *workers*. Isso pode evitar a execução de novas tarefas que podem ser inferidas a partir de diversas técnicas, como o uso de *Machine Learning* (ZHAO, 2011). Note que nem todos os tipos de tarefas podem ter suporte de inferências para serem concluídas. Para isto o criador do *workflow* da aplicação a partir de seu conhecimento no domínio do problema deve identificar quando isto pode ser feito.

Como exemplo de uma inferência a partir de uma relação matemática, pode ser apresentado o problema de sincronização de vídeos. Ele consiste em processar um conjunto de vídeos e sincronizá-los. Dentro deste conjunto é esperado que diversos vídeos possuam *overlaping*, ou seja, possuam trechos que foram filmados em paralelo por duas ou mais câmeras. Para um melhor entendimento, considere que neste conjunto de vídeos, há três vídeos ***A***, ***B*** e ***C*** que possuem *overlaping* entre si.

A tarefa do *worker* é assistir dois vídeos e identificar o ponto do vídeo em que ocorre este *overlaping*, sincronizando os dois vídeos. Para ***A***, ***B*** e ***C***, seriam necessárias no mínimo 3 *tasks* para solução do problema: uma para encontrar a relação ***AB***, outra para ***AC*** e uma final para ***BC***. Após contribuições iniciais duas destas tarefas convergiram: ***AC*** e ***BC***. A partir destas duas relações é possível encontrar AC, ou seja, a partir das duas primeiras tarefas convergidas é possível inferir a última tarefa, tornando-a também convergida e finalizando todas tarefas. De modo geral, o algoritmo de inferência consiste em uma busca de profundidade entre

as tarefas já convergidas. A profundidade que a busca pode atingir pode limitar o processamento em tempo real, e também pode gerar a propagação de erros entre tarefas e inferências, mas em contrapartida pode gerar um número maior de inferências.

Através do aprendizado de máquina, é possível utilizar um subgrupo de tarefas convergidas através da contribuição de *workers* para treinar uma solução. Essa solução em seguida será capaz de inferir o restante das tarefas que não foram encerradas. Um exemplo dessa abordagem é apresentado por Zhao *et al.* (ZHAO, 2011). Em sua proposta, ele usa as contribuições da *crowd* para treinar sua solução de reconhecimento automático de atividades humanas em vídeos.

5.5.3. Armazenamento de Contribuições

As contribuições geradas pelos *workers* precisam ser armazenadas em um repositório para pós-processamento. Além dos valores das contribuições, é possível armazenar todo o histórico da contribuição: cada tarefa realizada por cada usuário, seu resultado, o tempo gasto, recompensa gerada, e tudo que for desejado.

Estes dados são utilizados pelo algoritmo de convergência para verificar se as tarefas convergiram, e se é possível encerrar a execução da tarefa. A partir da análise das contribuições, são gerados os artefatos de saída que possuem a solução para o problema original: os tempos de sincronização entre os vídeos; as anotações realizadas; as notas alcançadas na avaliação de qualidade e outros.

5.6.CONSIDERAÇÕES

Seja de forma direta para resolver um problema, seja de forma indireta, para oferecer valores de entrada para o treinamento de soluções automáticas, o *crowdsourcing* tem sido usado tanto pelas empresas, quanto pela academia. Este capítulo fez uma análise de como o *crowdsourcing* pode ser integrado à solução de problemas do domínio de vídeos. Com base no conhecimento adquirido, foi proposto o *CrowdVideo*, que explicita as principais necessidades dos sistemas que envolvem o processamento de vídeo usando uma *crowd*.

Além de servir como base para as etapas de simulação descrita a seguir no Capítulo 6 e de fundamentar as aplicações desenvolvidas e usadas nos

experimentos do Capítulo 7, o modelo usado para a implementação de duas plataformas: a plataforma *LiveSync* (SEGUNDO, DE AMORIM e SANTOS, 2016), para sincronização de streamings de vídeo ao vivo, e a *CrowdNote*, uma plataforma para anotação de vídeos (AMORIM, SANTOS, *et al.*, 2017)

6. SIMULAÇÃO DA SINCRONIZAÇÃO PELA CROWD

Este capítulo tem como objetivo apresentar e analisar os resultados de uma simulação envolvendo o uso da *crowd* para solucionar o problema de sincronizar vídeos correlacionados. A simulação também visa avaliar a relação entre diversos aspectos da *crowd* e os resultados produzidos por ela, em termos de quantidade e qualidade das contribuições dos *workers* para a solução de um problema.

Apesar de compartilhar diversas aspectos em comum com o *CrowdVideo* apresentado no capítulo 5, o simulador (Figura 19) introduz uma modificação importante: o *worker-space* (parte responsável pela implementação do espaço de execução das tarefas pelos *workers*) é substituído por um componente que simula o comportamento da *crowd* recrutada para sincronizar os vídeos. Este componente é composto por (i) um simulador de *workers* que cria instâncias de membros da *crowd* e (ii) um simulador de respostas para estes *workers*. Cada *worker* pode possuir um grau de confiança diferente, baseado em seu perfil. Os perfis atribuídos a um *worker* são aqueles definidos por Kazai (2011): *diligents*, *competents*, *incompetents*, *sllopies* and *spammers.*

A Tabela 2 caracteriza cada perfil considerado na simulação, incluindo seu intervalo de precisão.

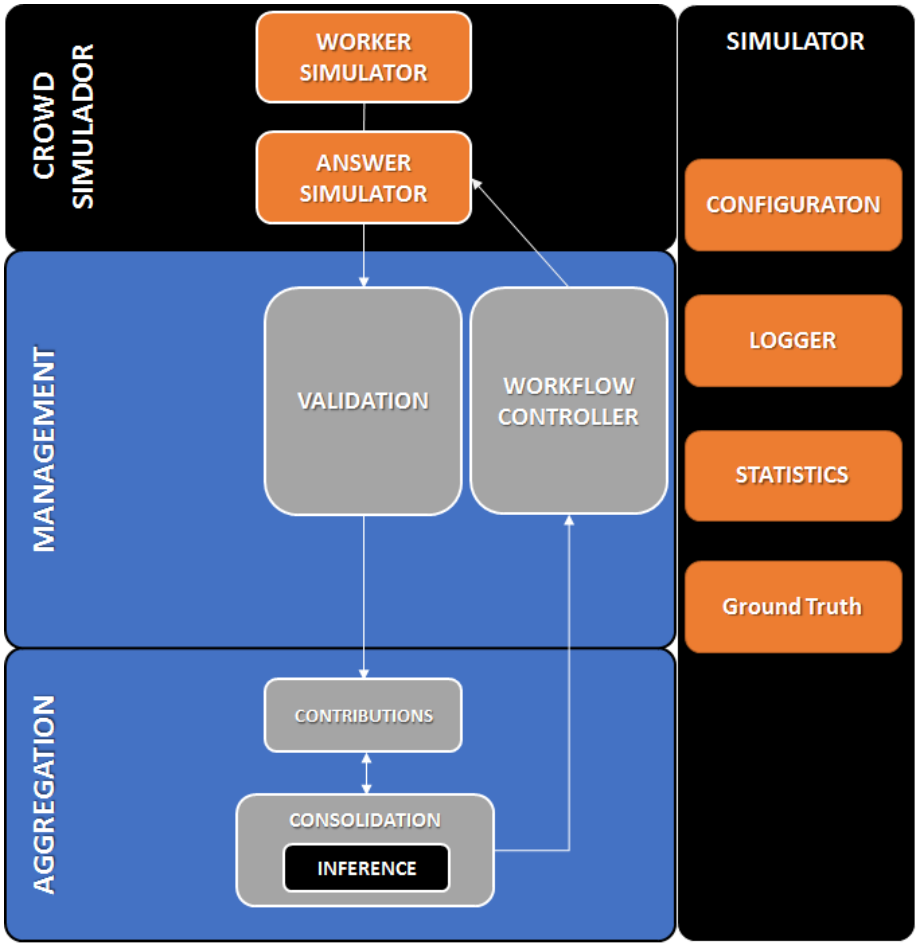
Na simulação, as contribuições geradas após a realização de cada tarefa levam em consideração: (i) a probabilidade de que um certo perfil de *worker* tenha sido escolhido e (ii) a probabilidade da contribuição gerada por aquele perfil seja válida ou não, baseado na sua acurácia.

Além das contribuições dos *workers*, a simulação permitiu configurar alguns parâmetros, tais como: (i) o número de simulações a serem realizadas; (ii) os parâmetros a serem passados aos componentes com o objetivo de testar diferentes valores de configuração, e (iii) quais os algoritmos utilizados na seleção e distribuição de tarefas, e o modelo de convergência das contribuições.

Na simulação, o componente ***Logger*** tem como função gerar os arquivos de saída, que servem de base para análise dos resultados. Todos arquivos gerados podem ser posteriormente usados pelo componente ***Statistics***, que agrupa os

outputs de cada teste, gerando médias, desvios e gráficos necessários para compreensão dos resultados obtidos.

Figura 19 - Modelo do simulador



Fonte: Elaborada pelo autor

Tabela 2 - Perfil dos Workers (KAZAI, 2011)

Perfil	***Características***	***Precisão(%)***
Diligent	Precisos, resolvendo várias *tasks*, mas demorando um pouco mais de tempo;	[71,92]
Competent	Precisos, resolvendo várias *tasks*, mas em menos tempo;	[67,100]
Sloppy	Não se preocupam com qualidade, produzindo uma baixa acurácia;	[10,51]
Incompetent	*Workers* com falta de habilidade ou pouco entendimento da *task*;	[33,57]
Spammer	Imprecisos. Composto normalmente membros maliciosos;	[0,33]

Por fim, o ***Ground Truth*** (valores previamente conhecidos do resultado esperado) é utilizado de duas formas: (i) como input para o ***Statistics***, onde o resultado gerado na simulação será comparado ao ***Ground Truth*** para avaliar a precisão das respostas geradas; e (ii) serve de base para geração das respostas

corretas geradas pela *crowd* simulada. Quando um membro da *crowd* é selecionado para gerar uma tarefa, o ***Ground Truth*** fornece o(s) valor(es) de base para determinar se a contribuição gerada pelo *worker* é válida ou não.

O ***Ground Truth*** pode ser gerado a partir de dados de um problema real, cujo resultado é conhecido, sendo possível comparar uma instancia real, com as respostas geradas pelo simulador, ou é possível gerar instâncias aleatórias, permitindo análise de casos diversos que podem não estar presentes em um ***Ground Truth*** baseado em um problema real.

6.1.Método

A simulação considera um cenário de sincronização de múltiplos vídeos, no qual a tarefa de um *worker* se resume a receber dois vídeos completos, dentre um conjunto de *N* vídeos possíveis, e a indicar um ponto de sincronização entre eles, caso exista. O *workflow* (descrição das atividades a serem realizadas) para solução do problema proposto é:

1. O algoritmo de distribuição de vídeos envia aos *workers* um par de vídeos a ser analisado;
2. O *worker* analisa os dois vídeos, e identifica se há entre eles um ponto de sincronização. Se houver, a sua contribuição será constituída de um número que representa a diferença de tempo entre o início de cada vídeo. A partir desse tempo é possível apresentá-los de forma síncrona:
    a. Esse valor pode ser preciso ou impreciso, de acordo com a confiabilidade do perfil do *worker* utilizado para aquela contribuição. A partir da precisão do perfil do *worker*, uma contribuição correta ou errada será gerada;
    b. Quando uma contribuição correta ocorre, o *Ground truth* é utilizado como base para a contribuição. O valor esperado é encontrado no *Ground truth*, e em seguida é adicionado o *bias* (valor de imprecisão), que varia de acordo com a configuração do simulador;
3. A nova contribuição é analisada, e é definido se a tarefa convergiu ou não:
    a. A tarefa converge se a probabilidade de uma resposta atingir o limite de convergência superior, ou se todas as respostas estiverem com probabilidade menor que um limite de convergência inferior;

   b. Se a tarefa convergiu, não há mais necessidade de contribuições sobre ela, se não, uma nova contribuição será requisitada;
4. Após uma nova convergência, a inferência é acionada de forma a inferir novos resultados a partir dessa mudança;
   a. Tarefas inferidas, não precisam receber novas contribuições;
5. Quando todas as tarefas convergirem, ou forem inferidas, o problema é considerado solucionado. Dessa forma, serão conhecidas todas as *N–1* relações temporais entre todos os *N* vídeos do repositório a ser sincronizado.

Neste cenário, é considerado o processamento *online* das contribuições, para assim analisar o impacto das variáveis no número de contribuições necessárias. Para cada valor de variável são executadas 100 instâncias (HUNG e AL., 2013) da simulação com o mesmo valor da variável, a fim de evitar erros na análise. A partir da média e variação destes valores, pode-se identificar a influência de cada variável no problema e assim apresentar considerações sobre.

6.2. Variáveis de avaliação

A simulação tem como objetivo identificar o impacto das seguintes variáveis na solução do problema de sincronização:

- Limites de convergência: delimita os valores de probabilidade *Pi* para que as relações entre os vídeos convirjam ou divirjam. Os valores de *Pi* definem se as contribuições resultantes da execução das tarefas convergiram para um valor esperado;
- Confiabilidade dos membros da *crowd*: a confiabilidade irá variar de acordo com a quantidade de usuários confiáveis e não confiáveis semelhante à classificação de (YU, SHEN, *et al.*, 2012). *Workers* não confiáveis são aqueles cuja precisão média é menor que 50%, e os confiáveis, os que possuem precisão maior que 50%. Sendo assim: *spammers*, *sloppies* e *incompetents* são considerados não confiáveis, e *competents* e *dilligents* confiáveis;
- Intervalo considerado para agregação de valores: as contribuições devem ser agrupadas para gerar as categorias de contribuições de acordo com a distribuição categórica utilizada no algoritmo de convergência. Assim, esse

valor corresponde ao intervalo em que diferentes contribuições são consideradas como uma mesma categoria. Quanto maior o intervalo, menor é a precisão exigida para que uma contribuição seja considerada válida e mais rapidamente um agrupamento cresce; porém, menor será a precisão alcançada na solução do problema;

- Profundidade da busca de inferência: quanto maior o nível de profundidade definido para a busca, maior a probabilidade de que o sistema possa inferir uma contribuição. Por outro lado, maior será o custo computacional para a busca e a possibilidade de propagação de erros na geração da solução final do problema;
- Algoritmo de distribuição de tarefas: o algoritmo de distribuição, em conjunto com o de convergência, é essencial para a distribuição de tarefas, e definição da quantidade de atividades a serem realizadas;
- *Bias*: é um valor de imprecisão que pode ser adicionado à contribuição de cada *worker*, para simular o erro humano na execução da tarefa.

6.3.Métricas de avaliação

Tendo definido as variáveis, precisa-se definir como avaliar o impacto da mudança de seus valores na simulação. Para avaliação destes testes são utilizadas as seguintes métricas:

- Número de contribuições: o número de contribuições necessárias para atingir a convergência ou divergência de todas as tarefas. Para cada simulação o número de contribuições necessárias para realização de todas as tarefas, e consequentemente, para solução final do problema pela *crowd* é obtido;
- Convergência: explicita o número de tarefas para as quais as contribuições convergiram para valores esperados e as que convergiram para valores errados. A definição de uma convergência correta ou incorreta é feita a partir da comparação com o *Ground Truth*. Além das convergências, de forma complementar, as tarefas que divergiram são contabilizadas;

- Inferências: a quantidade de valores inferidos corretamente (e incorretamente) a partir dos resultados de tarefas concluídas;
- Tempo de simulação: a métrica de tempo será utilizada para verificar se a variação das variáveis de avaliação tem impacto no tempo de processamento total das tarefas simuladas;
- Precisão da solução: mede qual a porcentagem de tarefas convergidas, de acordo com o *Ground T*ruth. Tarefas convergidas de forma correta são aquelas cujo valor encontrado possui um erro menor à 0,5s quando comparado ao *Ground Truth*. Já tarefas convergidas de forma errada possuem uma diferença maior que 0,5s;

6.4. Ambiente de Testes

O ambiente de simulação foi executado em computadores com a seguinte configuração: DELL, core i5-3570 @ 3.40GHz 8GB de memória com sistema operacional Ubuntu 14.04. Todos componentes foram desenvolvidos em *JavaScript* para execução no NODE.JS.

6.5. Experimento

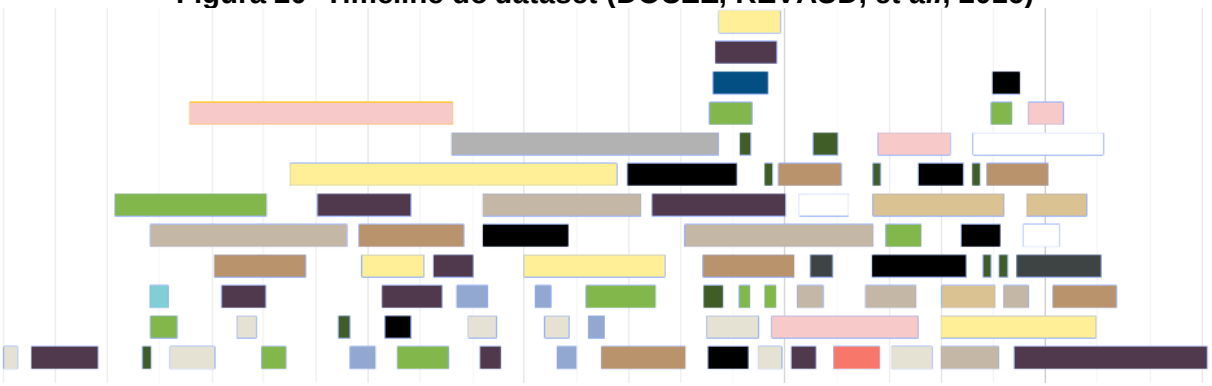
A seguir são detalhadas as simulações de cada variável. Como o intuito é avaliar cada variável isoladamente, cada experimento varia apenas em termos da variável de avaliação que está sendo testada. Como padrão, os demais valores utilizados são:

- Limite superior de convergência: 51%
- Limite inferior de convergência: 20%
- Confiabilidade da *crowd*: 60%
- Intervalo de agrupamento: 2s
- Profundidade da busca de inferência: 10
- Algoritmo de distribuição: lista de prioridade
- Bias: 0,2s

Estes valores foram propostos a partir de testes empíricos preliminares para avaliar o correto funcionamento dos algoritmos e de outros experimentos realizados, servindo então de base. Quando necessário, de acordo com o experimento, a mudança destes valores é informada.

Outro fator importante para simulação é a quantidade de vídeos a serem utilizados. O *dataset* escolhido (DOUZE, REVAUD, *et al.*, 2016) possui 89 vídeos que possuem diversos pontos de sobreposição, ideal para o estudo de caso de sincronização de vídeos. A distribuição dos 89 vídeos sobre um eixo de tempo comum pode ser vista na Figura 20, onde cores iguais representam vídeos de filmados a partir de uma mesma câmera.

Figura 20 -Timeline do *dataset* (DOUZE, REVAUD, *et al.*, 2016)



Fonte: Elaborada pelo autor

O tamanho total deste *dataset* é de 22.871,84s (soma do tempo de todos os vídeos), distribuídos em um intervalo de 4.327,3s (tempo desde o início do primeiro vídeo, ao final do último considerando todas as sobreposições). Existe neste *dataset* um total 3.916 relações possíveis entre os vídeos. Ou seja, 3.916 pares de vídeos podem ser entregues aos *workers* para que eles executam a tarefa de identificar pontos de sincronização ou se não existe sincronização entre os vídeos da tarefa.

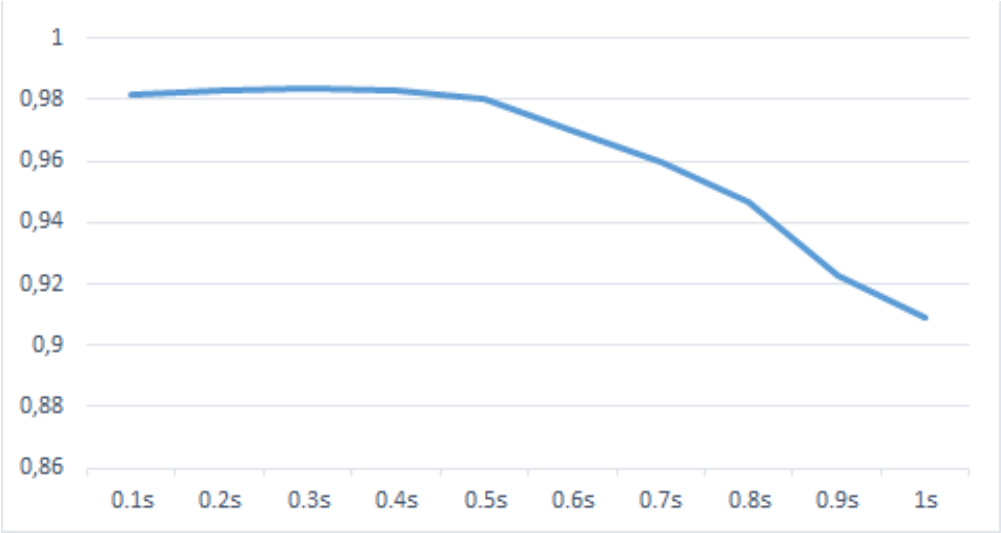
6.5.1. Bias

O Bias pode, por exemplo, ser o valor de erro entre duas contribuições na anotação em um mesmo instante de vídeo, porém, as anotações são feitas em tempos ligeiramente diferentes, ou seja, pode ser considerado o fator de erro humano na execução da tarefa. Essa é uma característica ligada ao tipo de conteúdo contínuo anotado, diferente de outras tarefas em processos *crowdsourcing* como a anotação de *tags* em imagens.

Para simular possíveis imprecisões das contribuições dos *workers* efetivas criadas a partir do *Ground Truth*, cada contribuição produzida recebe um valor de incerteza que pode variar em um intervalo de ±*β*. O valor de *β* em cada instância nas simulações executadas vai de 0,1s a 1,0s. Como exemplo, se um *worker* define um ponto de sincronização no instante 10,3s e o bias para ±0,2s, o valor da contribuição considerado na simulação pertencerá ao intervalo [10,1; 10,5]s.

A Figura 21 mostra a piora da precisão da solução a medida que o bias aumenta. A precisão é medida a partir da comparação de todos os valores obtidos após a realização das tarefas com os valores correspondentes esperados do *Ground Truth*. Com o intervalo variando em ±0,1s, a precisão obtida foi de 98%, porém com o aumento do bias, a precisão é reduzida, chegando a 90% quando a variável assume o valor de 1,0s.

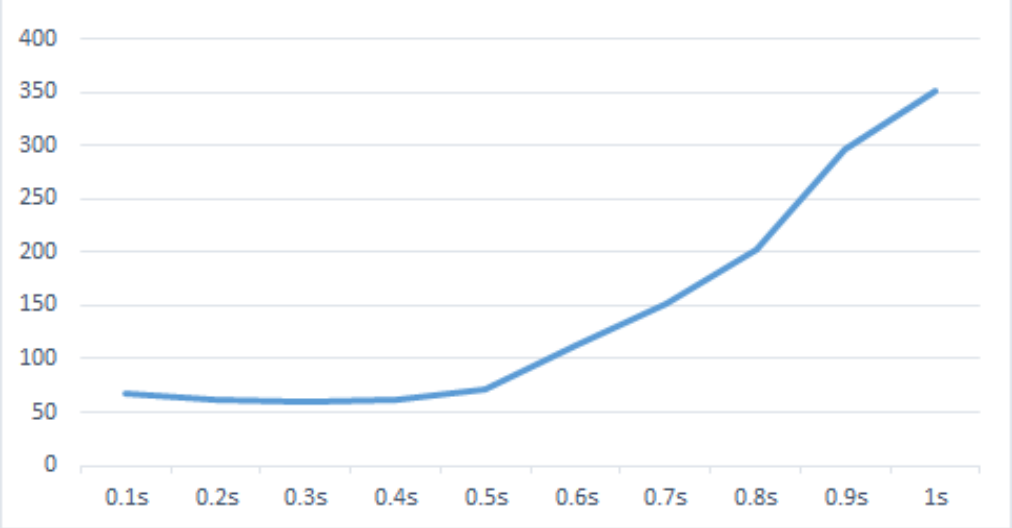
Figura 21 - Precisão vs. Variação do Bias



Fonte: Elaborada pelo autor

Os erros decorrentes da imprecisão dos valores dos instantes de sincronização anotados com relação aos esperados também têm impacto nos valores obtidos por meio de inferência. Inferências feitas sobre valores imprecisos produzem valores com erros e a propagação destes erros é inevitável pela própria forma iterativa do processo de inferências. O efeito do erro acumulado no processo de inferência é ilustrado no gráfico da Figura 22. Nota-se que o número de erros com β=±1s (352,07 em média) é quase 5 vezes maior do que com β=±0,1s (67,74, em média).

Figura 22 - Erro na Inferência vs. Variação do Bias



Fonte: Elaborada pelo autor

6.5.2. Profundidade da Inferência

A profundidade da inferência está relacionada ao nível máximo que a busca pelo valor pode alcançar. Para inferir valores no contexto de sincronização, é possível utilizar um algoritmo de busca. Neste algoritmo os níveis da busca são as relações indiretas entre os vídeos, por exemplo, em um dataset com cinco vídeos A, B, C, D e E é possível inferir a relação [A,E] a partir da transitividade entre todos os vídeos {[A,B], [B,C],[C,D],[D,E]}.

Quanto maior o nível, mais caminhos para inferir relações entre tarefas podem ser utilizados. O maior nível na busca da inferência que pode ser alcançado é através de uma relação indireta que passa por todas as outras relações, neste caso, uma inferência que precisa analisar os outros 88 vídeos. No outro extremo.

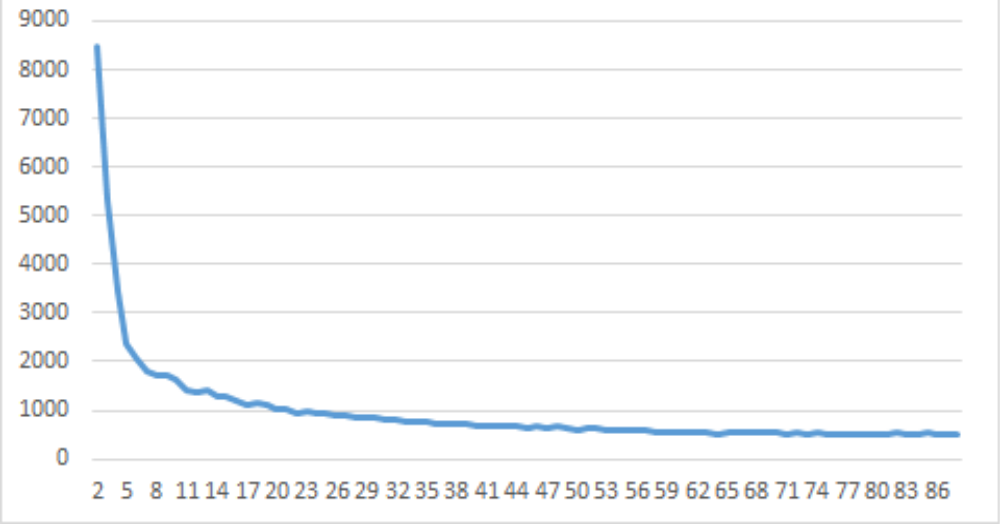
Antes de comentar os resultados obtidos, se faz necessário explicar os casos com profundidades 0 e 1. Nestes casos, a busca não pode inferir nenhum valor pois não há caminhos ligando as tarefas convergidas: no nível 0 não há caminhos e no nível 1 só há caminhos de um vídeo para ele mesmo. Em outras palavras, são casos onde nenhuma inferência foi realizada, sendo necessário a convergência de todas tarefas a partir de contribuições diretas dos *workers*. Isto gera na curva de análise uma grande distorção. Sendo assim os casos de nível 0 e 1 serão sempre comentados a parte nos gráficos apresentados.

Os valores que sofrem mudança significativa com a mudança do nível de inferência são: o número de contribuições, o tempo de execução da simulação, e as quantidades de convergências e inferências.

Sem nenhuma inferência realizada (valores 0 e 1 para esta variável), a simulação mostrou que o número médio de contribuições necessárias para resolver todas as tarefas é de 15.518. Porém, à medida que o nível da inferência aumenta, a quantidade de contribuições necessárias diminui (com apresentado na Figura 23). Ela passa de 8.464,95 em média no nível 2 para 523,03 no nível 88. O número de contribuições alcança a centena 5 no nível 57 de inferência, com 577 contribuições necessárias.

O tempo de execução é diretamente relacionado ao número de contribuições e tempo de processamento dos algoritmos. Sendo assim, quanto maior o número de contribuições requisitadas para a solução geral do problema, maior é o tempo de processamento esperado. Isto ocorre, por exemplo, com as profundidades 0 (nenhuma inferência) e 1 (apenas 1 nível de inferência), que demoraram em média 58,7s. Esse valor foi 69% superior ao processamento mais lento quando ocorre a inferência: no nível 2.

Figura 23 - Número de Contribuições vs. Nível de Inferência

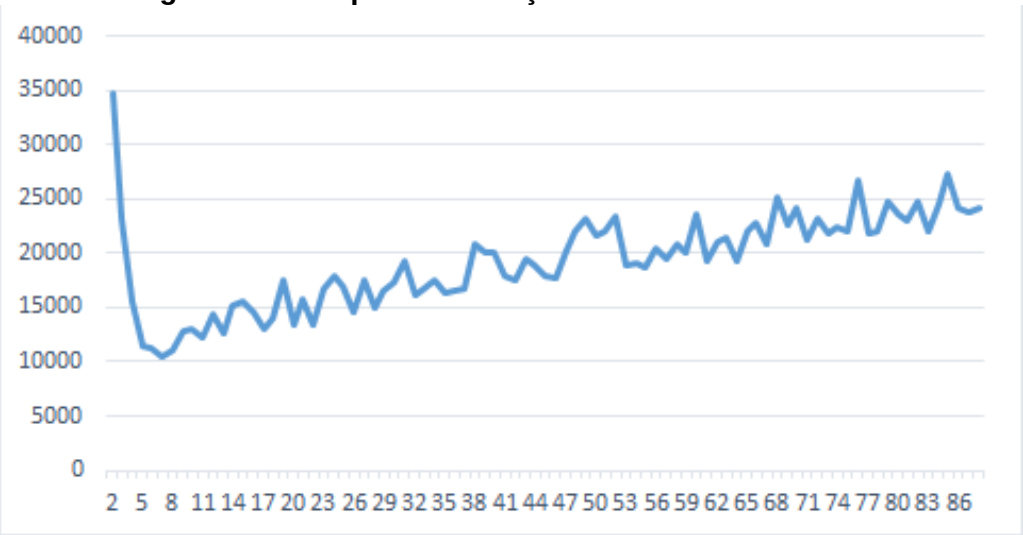


Fonte: Elaborada pelo autor

Porém, observando o gráfico da Figura 24, percebe-se o aumento gradual do tempo, mesmo sabendo que o número de contribuições reduz. Isto acontece devido ao algoritmo de inferência, pois com um nível de profundidade maior, acaba gastando mais tempo na tentativa de inferir novas relações.

Em relação ao número de convergências, como no início há poucas inferências realizadas, existe uma quantidade maior de convergências a partir das contribuições diretas (Figura 25). E com uma quantidade maior de convergências diretas, ocorreu uma quantidade maior de erros.

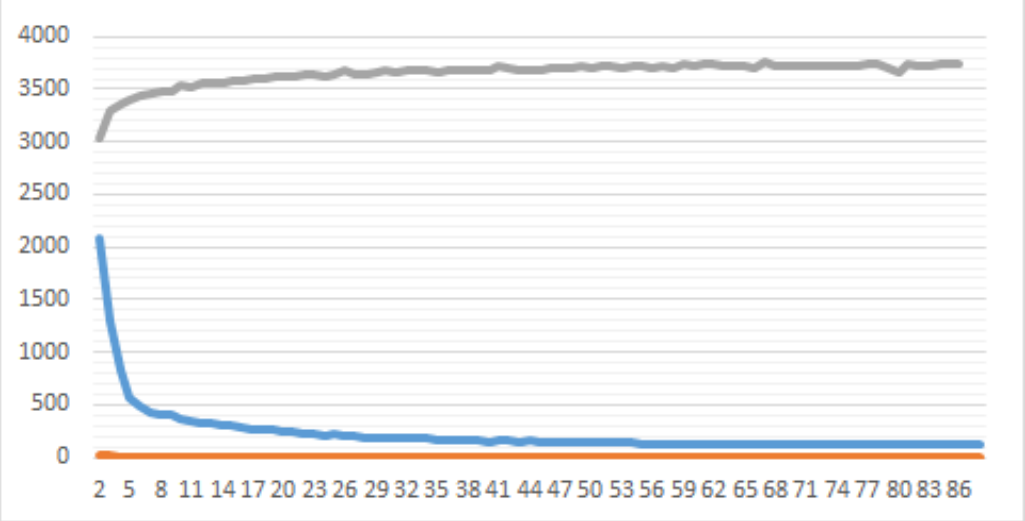
Figura 24 - Tempo de Execução vs. Nível de Inferência



Fonte: Elaborada pelo autor

No nível 2, houve um total médio de 1054 convergências, sendo que 1,8% destas obtiveram um valor errado, quando comparado ao valor do *Ground Truth*. A quantidade de convergências erradas reduz à medida que o nível aumenta, chegando a 0,95% no nível 82. Em contrapartida, a quantidade de inferências aumenta de acordo com o aumento dos níveis.

Figura 25 - Convergência e Inferência vs. Nível de Inferência. Em Cinza: inferências corretas; Azul: convergências corretas; laranja: convergências erradas



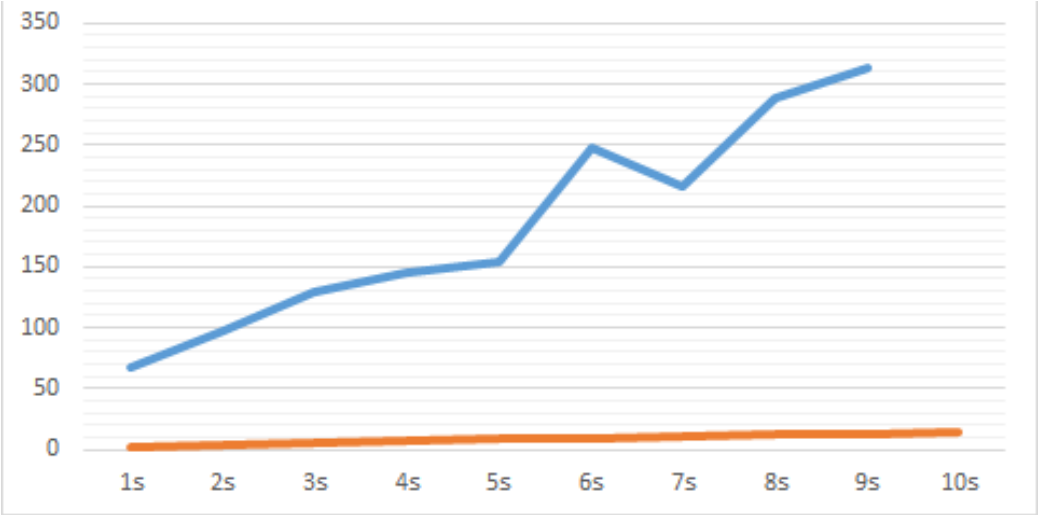
Fonte: Elaborada pelo autor

6.5.3. Intervalo de Categorização

O intervalo de categorização é utilizado quando se aplica uma distribuição categórica para definir se uma tarefa converge ou não em uma das possíveis categorias. No cenário de testes as categorias são os intervalos de tempo nos quais as contribuições são agrupadas. O número de categorias é indefinido, variando de acordo com as contribuições geradas.

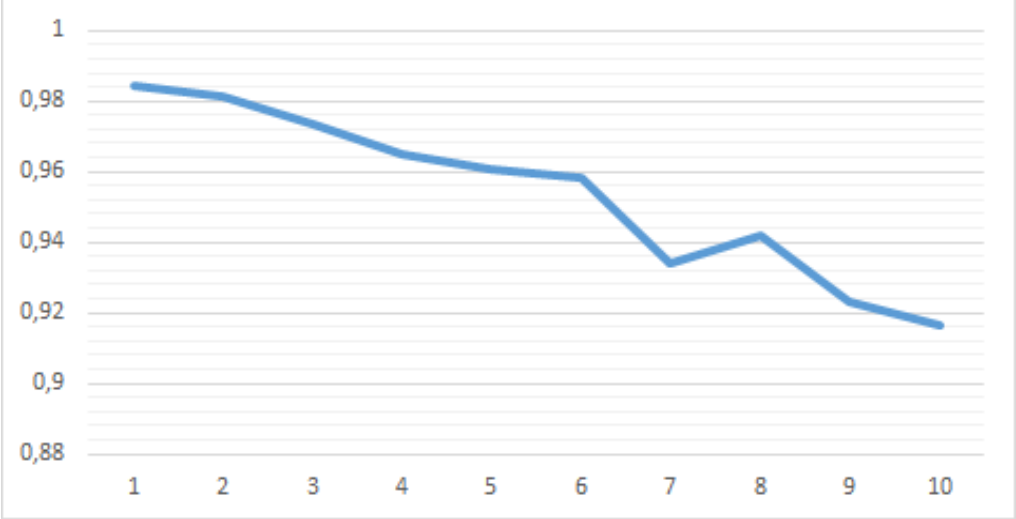
A fim de verificar a influência dessa variável, o valor do intervalo foi definido de 1s à 10s. A variação do intervalo considerado para agregação de valores mostrou impacto no número de convergências, inferências e precisão. Mais especificamente, nas falsas convergência e inferências, que não deveriam ter sido encontradas. Como pode ser visto na Figura 26, quanto maior o intervalo, maior a quantidade de erros gerados, e com isso, uma redução da precisão (Figura 27).

Figura 26 - Convergência e Inferência vs. Intervalo Categorização. Em Azul: inferências erradas; Laranja: convergências erradas



Fonte: Elaborada pelo autor

Figura 27 - Precisão vs. Intervalo de Categorização



Fonte: Elaborada pelo autor

6.5.4. Algoritmos de distribuição de tarefas

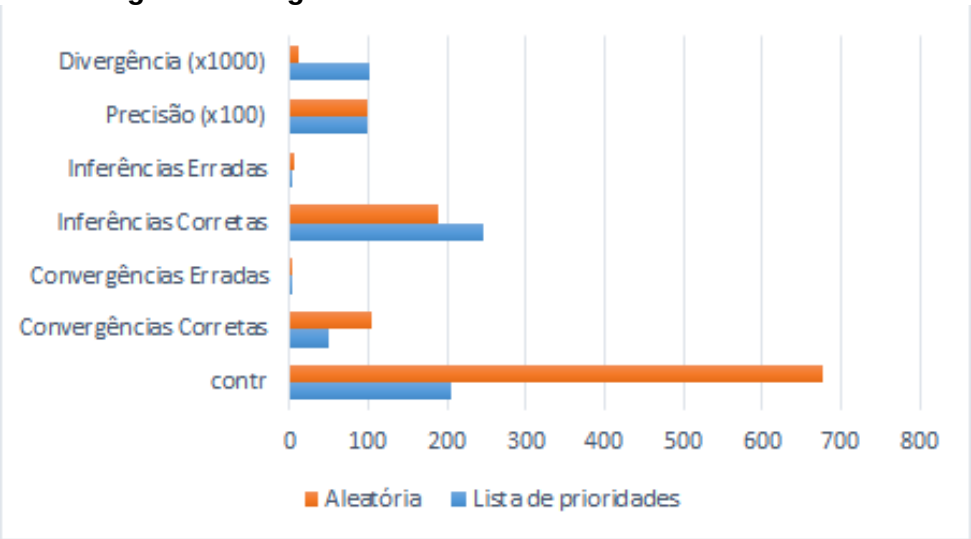
O algoritmo de distribuição é responsável por distribuir as tarefas entre os *workers*, de forma a tentar minimizar a quantidade de contribuições necessárias para a solução do problema. Para verificar o comportamento da distribuição, foram desenvolvidos dois algoritmos para os testes: o baseado na lista de prioridades (apresentado na seção 5.1.3.1) e um totalmente aleatório.

Na simulação, porém, foi necessário reduzir o número de vídeos utilizados, de 89 para 25, uma vez que com 89 vídeos não foi possível alcançar, em tempo hábil, a solução do problema (encontrar todos os *N* pontos de sincronização entre os vídeos

da base) com o uso do algoritmo de distribuição aleatória. Um indicio claro de que é necessária alguma inteligência na distribuição de tarefas.

A Figura 28 apresenta um gráfico comparativo entre as duas abordagens. Nele é possível observar que a lista de prioridades é superior para todas as métricas, já que possui uma menor quantidade de divergência, erros de convergência e inferência, uma quantidade menor de contribuições e uma precisão um pouco melhor (0,4%).

Figura 28 - Algoritmo Aleatório vs. Lista de Prioridades



Fonte: Elaborada pelo autor

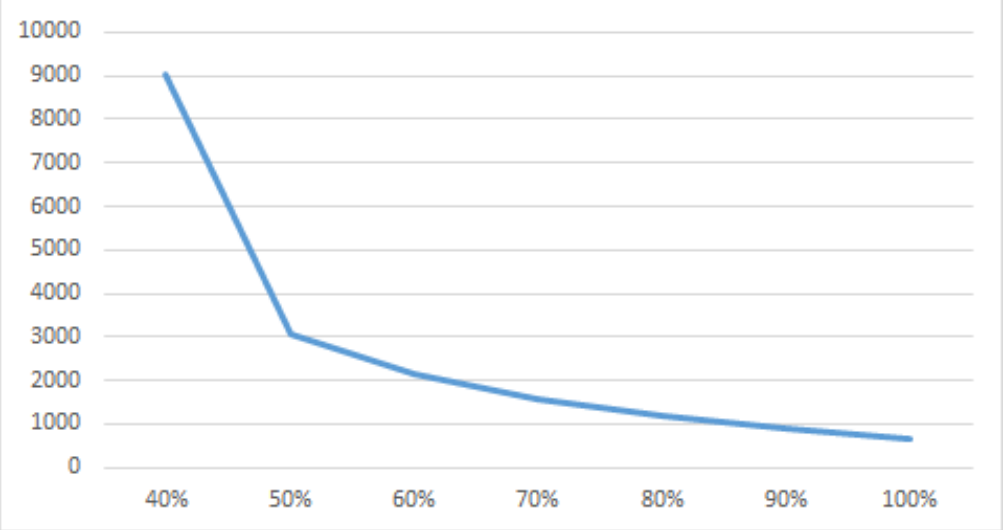
6.5.5. Confiabilidade da *Crowd*

A confiabilidade corresponde a taxa de *workers* que são considerados confiáveis e não confiáveis. No experimento as taxas variaram de 40% a 100%. Abaixo de 40%, a simulação se tornava inviável pela alta quantidade de contribuições necessárias para chegar à solução do problema (acima de 100 mil contribuições). Na simulação, de forma complementar, à medida que o número de *workers* confiáveis diminui, o número de não confiáveis aumenta.

Como esperado, a variação da confiabilidade tem impacto significativo em todas as métricas de avaliação.

A Figura 29 mostra a redução do número de contribuições quando aumentado o grau de confiabilidade da *crowd*.

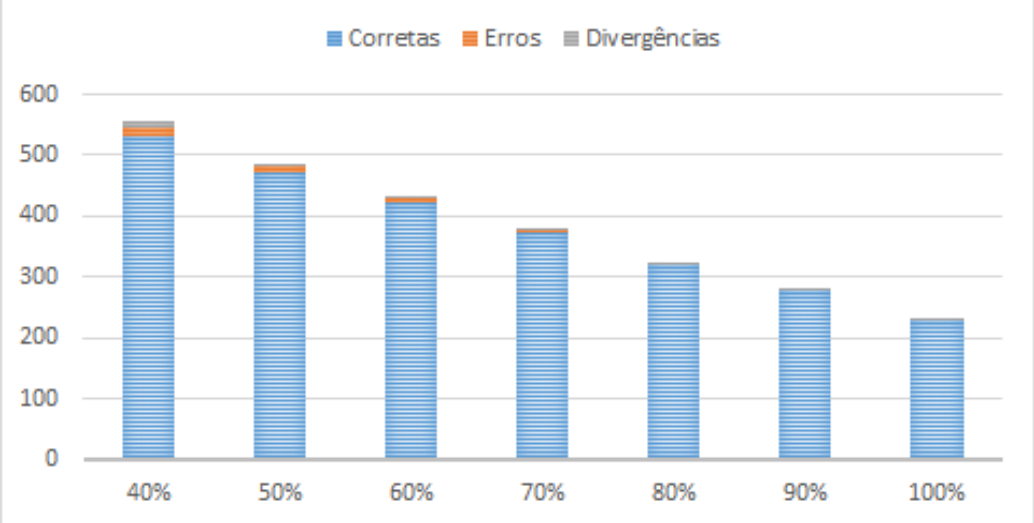
Figura 29 - Contribuições vs. Grau de Confiança



Fonte: Elaborada pelo autor

Com o aumento da confiabilidade também é possível notar a redução da quantidade de convergências diretas (aquelas que são descobertas a partir das contribuições dos *workers*, sem nenhuma inferência) necessárias para obter uma resposta (contribuição válida), e também a redução do número de erros (Figura 30).

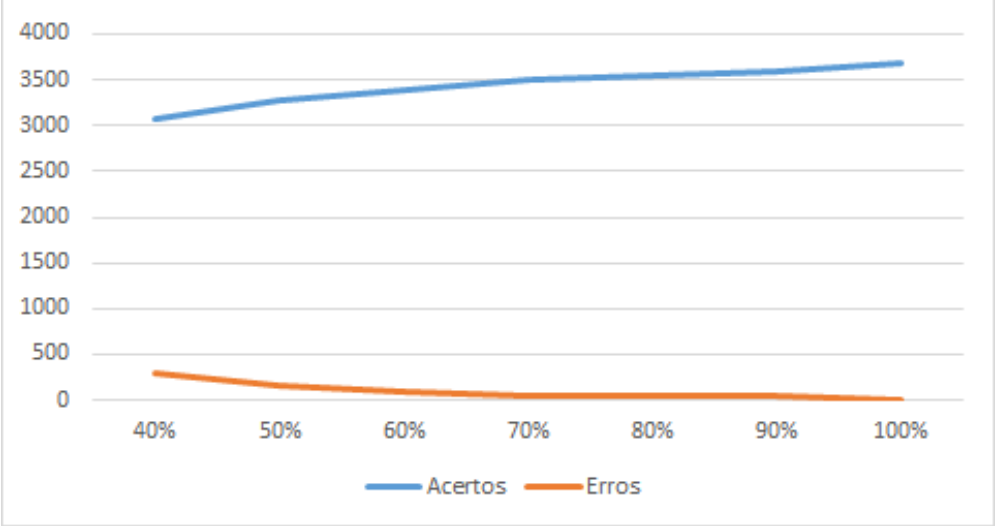
Figura 30 - Convergência e Divergência vs. Grau de Confiança



Fonte: Elaborada pelo autor

A redução da necessidade de convergências diretas é justificada pelo número de inferências que são realizadas (Figura 31), que aumenta junto com a confiabilidade da *crowd*, e uma *crowd* mais confiável, reduz a quantidade de erros na inferência.

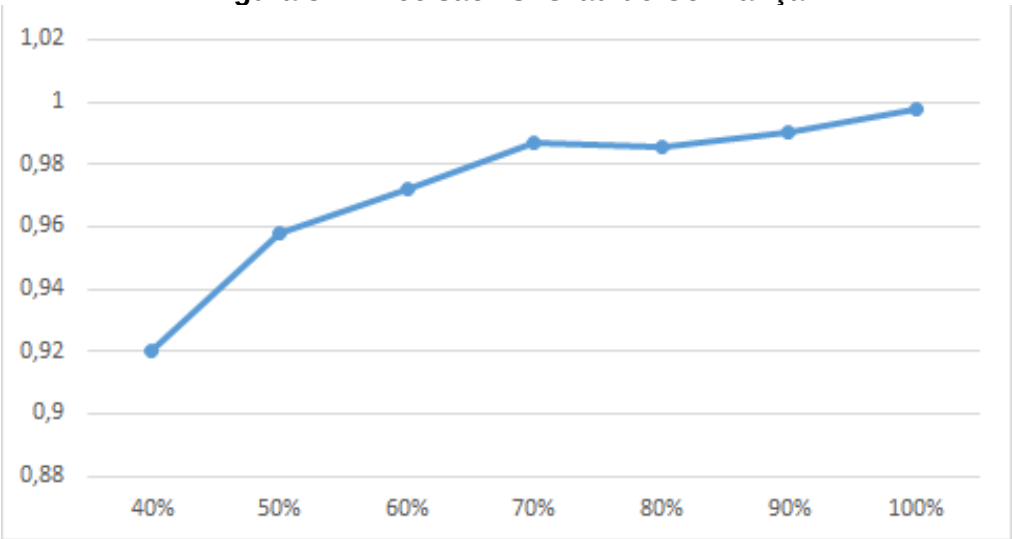
Figura 31 - Inferência vs. Grau de Confiança



Fonte: Elaborada pelo autor

Por fim, com a redução da quantidade de erros, observar-se um aumento na precisão da solução como pode ser visto na Figura 32.

Figura 32 - Precisão vs. Grau de Confiança



Fonte: Elaborada pelo autor

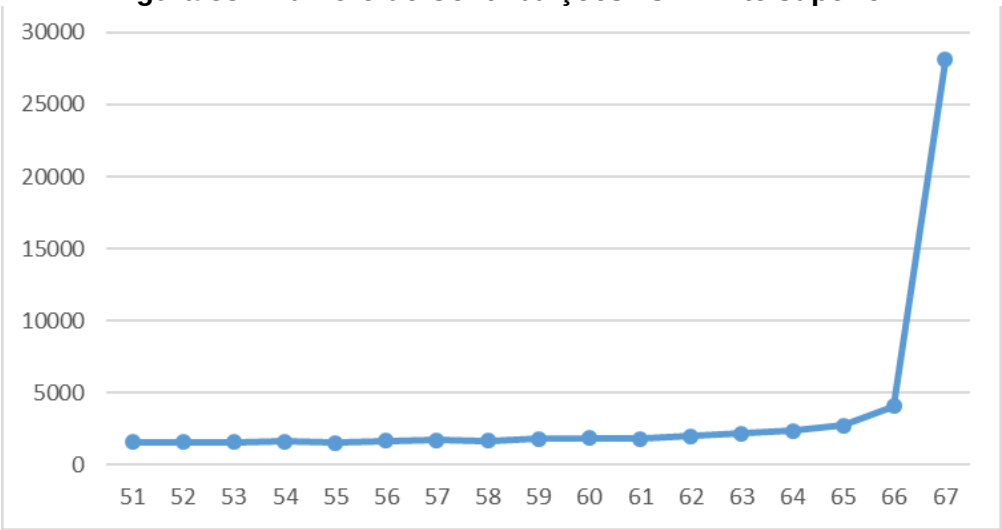
6.5.6. Limite Superior

O limite superior marca o ponto que uma probabilidade na distribuição categórica deve atingir para que a tarefa seja considerada convergida para aquela categoria (valor do *offset*). Esse limite é variado de: 51% a 100%. 51% foi escolhido como menor valor pois necessita de pelo menos duas contribuições para convergir. O outro extremo, 100% é o maior valor possível para a variável.

Porém, durante a simulação ocorreu um máximo do limite superior em 67%. Valores maiores que estes exigem um valor muito grande de contribuições, tornando

a simulação impraticável. A Figura 33 mostra os resultados obtidos. Nela, observa-se uma pequena variação do limite entre 51% e 61%, quando lentamente a curva começa a subir, tomando valores muito acima dos demais, ao chegar em 67%.

Figura 33 - Número de Contribuições vs. Limite superior



Fonte: Elaborada pelo autor

6.5.7. Limite Inferior

O limite inferior marca o ponto em que, caso todas as probabilidades da distribuição estejam abaixo desse valor, a tarefa é considerada divergida. A princípio, mesmo variando este valor entre 10% e 50%, não foi detectado nenhuma mudança significativa nas métricas.

6.6.CONSIDERAÇÕES

A simulação teve como objetivo evidenciar as diferentes variáveis que poderiam ter impacto no comportamento do processo de sincronização proposto. Foi observado que os principais fatores que influenciam são: a qualidade da *crowd*, o uso ou não de um algoritmo de inferência, o intervalo de agrupamento ao se usar a categorização e o algoritmo de distribuição de tarefas aos *workers*. Estes fatores implicam diretamente na quantidade de contribuições que são necessárias à resolução do problema e na precisão alcançada ao final. Por exemplo, um aumento do Bias reduz a precisão da solução e gera mais erros de inferência; já um aumento dos níveis de inferência reduzem drasticamente o número de contribuições necessárias; mas se aumentarmos o intervalo de categorização, teremos a redução da precisão; por outro lado, se tivermos uma *crowd* mais confiável, termos impacto

positivo na precisão, redução de erro no uso da inferência, e também redução do número de contribuições necessárias; um último exemplo de resultado observado é o impacto do algoritmo de distribuição de tarefas: um algoritmo inteligente é capaz de reduzir o número de contribuições necessárias e aumentar o número de inferências realizadas.

7. EXPERIMENTOS

Ao longo deste trabalho foram descritas múltiplas técnicas e possibilidades do uso de *crowdsourcing* na sincronização de vídeos. Neste capítulo são detalhados os resultados de três dos principais experimentos realizados durante a pesquisa com a participação de *workers*. O primeiro experimento utiliza um sistema desenvolvido com a abordagem baseada em segmentos. O segundo, um experimento com uma abordagem híbrida, que combina o uso da *crowd* com sistemas automáticos. Por fim, no último experimento, a *crowd* é recrutada por meio de uma plataforma comercial de *crowdsourcing* (*MicroWorkers*) para realização das tarefas de sincronização.

7.1. CROWDSOURCING BASEADO EM SEGMENTOS DE VÍDEOS

O primeiro experimento utilizou a abordagem baseada em segmentos para sincronização dos vídeos. Para isto a aplicação coleta as contribuições de sincronização, e apresenta os vídeos e a interface de trabalho dos *workers* em uma interface web desenvolvida em HTML5/JS, se comunicando com um servidor NODE.JS que implementou a lógica de gerenciamento das tarefas.

Neste experimento, a *crowd* foi constituída de *workers* voluntários, tendo estes como recompensa apenas um lanche durante a execução das tarefas. Como todos os voluntários eram conhecidos, não foi necessário fazer nenhuma validação das contribuições em busca de contribuições maliciosas. De forma a minimiza o número de contribuições necessárias para resolução do problema, foi utilizada a distribuição de tarefas através de listas de prioridade para entrega das tarefas aos *workers*.

Para a agregação dos resultados foram utilizadas as técnicas de consolidação baseada na função de distribuição categórica das contribuições e o um limite superior de convergência (ver seção 5.5.1). De forma complementar, foi utiliza a inferência de dados de forma análoga ao exemplo descrito na seção 5.5.2.

Os principais conceitos associados à abordagem baseada em segmentos foram apresentados na seção 4.1.4. A avaliação da sincronização com essa abordagem foi feita através de um experimento controlado em laboratório. Um subconjunto do *dataset* de vídeo apresentado por Douze *et al.* (2016) foi usado

como entrada para o experimento de sincronização. Este *dataset* contém múltiplos vídeos capturados por um grupo de pessoas durante uma atividade de escalada. Os vídeos foram gerados por celulares, câmeras fixas e câmeras acopladas a capacetes. Movimentações das câmeras, tremulações e outros fatores associados ao processo de captura deste *dataset* de vídeos, o tornam suficientemente heterogêneo para ser considerado equivalente a um conjunto de *UGVs* correspondente a um evento.

Para a sincronização do *dataset*, Douze *et al.* (2016) utilizaram uma técnica automática, obtendo resultados não muito animadores. O método permitiu identificar de forma automática apenas 23% dos pares de vídeos que possuíam correlação direta entre seus conteúdos.

No caso desta tese, como se tratava de um experimento realizado em laboratório, um subconjunto de 9 vídeos foi selecionado para os testes da aplicação *crowdsourcing* desenvolvida. Como o número de membros disponíveis na *crowd* era limitado, foi necessário limitar também a quantidade de tarefas a serem realizadas. Assim, foram recrutados 9 membros do laboratório onde a pesquisa foi desenvolvida e 6 pessoas com habilidades no uso de tecnologias, num total de 15 *workers*.

Esses 9 vídeos foram então segmentados em trechos com duração de 5s, os quais foram enviados aos pares a cada um dos membros da *crowd* para que eles identificassem os pontos de correlação dos seus conteúdos. Não se pode afirmar neste ponto se este valor de intervalo de 5s é o ideal para o tipo de tarefa solicitada à *crowd*. Também não faz parte do escopo da tese determinar o valor ideal da duração dos segmentos, apesar de se saber de antemão que o esforço para realização das tarefas, assim como o sucesso em encontrar pontos de sincronização é fortemente dependente deste valor.

Em resumo, na proposta defendida nesta tese, os vídeos do conjunto de entrada são segmentados inicialmente em trechos de 5s.

Os resultados obtidos pela *crowd* puderam ser avaliados a partir do *ground truth* disponibilizado junto com o *dataset*, o qual tem uma tolerância de 0.5s para os valores de sincronização (DOUZE, REVAUD, *et al.*, 2016). A partir da comparação com o *ground truth*, foi possível verificar que a *crowd* conseguiu uma taxa de sucesso em encontrar os pontos de sincronização para 88% dos casos, enquanto

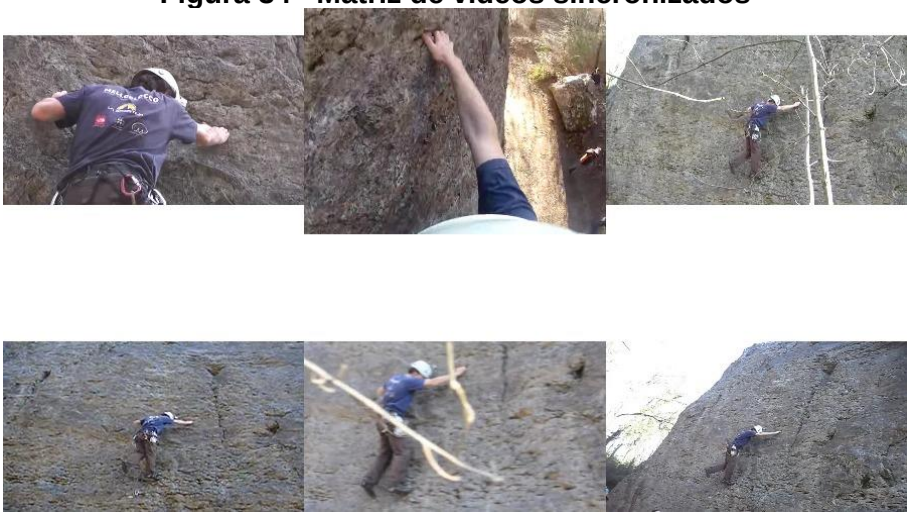
que os 12% restantes exibiram valores com erros superiores à tolerância de 0.5s, sendo considerados como insucessos.

Os 9 vídeos geraram 278 segmentos (1390s), os quais por sua vez resultam na combinação de 38503 possíveis relações. Porém, não foi necessário que os *workers* avaliassem todas as relações, na verdade foram necessárias “apenas” 1051 contribuições (comparações entre os pares) para atingir o resultado apresentado, envolvendo 421 relações (cada relação recebeu mais de uma contribuição). Isso se deve ao uso do algoritmo de inferência, que a cada nova contribuição é capaz de inferir várias relações, e com isso cortar o número de contribuições necessárias. Outro fator para este valor é que uma vez um vídeo sincronizado, não há necessidade de comparar seus segmentos com nenhum outro vídeo, removendo assim todas relações envolvendo os segmentos deste vídeo da lista.

Das 1051 contribuições, em cerca de 98% delas, os *workers* não conseguiram identificar um ponto de sincronização entre o par de segmentos analisados. Como adiantado na seção na seção 4.1.4 e comprovado no experimento, a maior parte das tarefas concluídas pela *crowd* na abordagem baseada em segmentos resulta em insucesso, isto é, na confirmação de que não há nenhum ponto de sincronização entre o par de segmentos analisado. Isto torna o uso desta técnica da forma implementada inviável.

A Figura 34 ilustra um instante no qual pode ser observada a correlação dos conteúdos de 6 vídeos provenientes de 6 diferentes câmeras. Este ponto de sincronização é resultante da combinação das contribuições da *crowd* durante o experimento. Os outros 3 vídeos do subconjunto do *dataset* não foram mostrados na figura porque não houve indicação pela *crowd* de que seus conteúdos estavam correlacionados aos conteúdos dos outros 6 vídeos da base. Através da figura, é possível observar a heterogeneidade do *dataset* e a sua similaridade ao tipo de vídeos considerados neste trabalho.

Figura 34 - Matriz de vídeos sincronizados



Fonte: Elaborada pelo autor

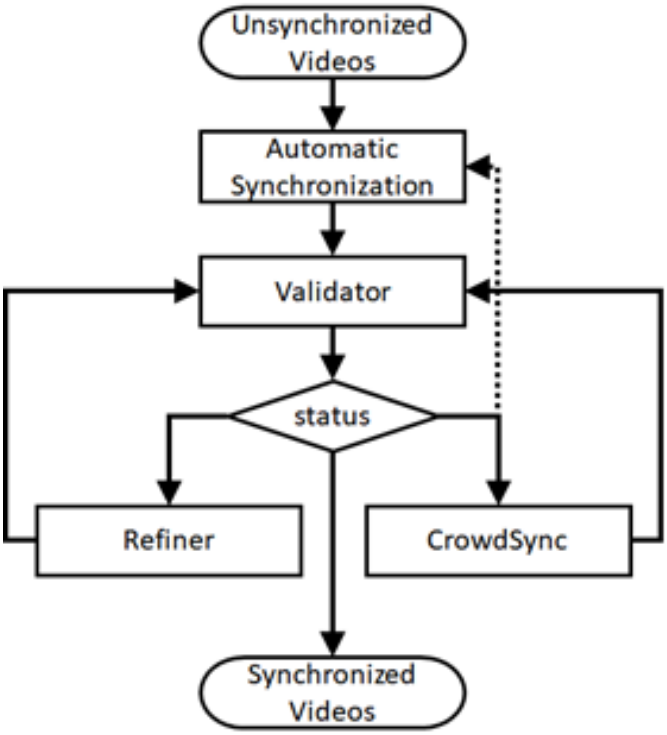
7.2.MÉTODO HÍBRIDO

Na abordagem híbrida, primeiramente os vídeos (entrada) são processados por um algoritmo automático (máquina), que gera resultados a serem validados e aprimorados pela *crowd* (humanos).

A Figura 35 mostra o *workflow* que especifica a sequência de atividades seguidas no modelo híbrido, sendo que sua entrada é um conjunto vídeos não sincronizados e produzidos de forma independente e descoordenada.

A primeira atividade no *workflow* corresponde à tentativa de sincronização automática dos vídeos a partir do processamento dos seus conteúdos. Neste ponto, não é possível garantir a qualidade da especificação gerada: no melhor cenário, todos os vídeos serão sincronizados corretamente, ou no pior caso, nenhum será sincronizado. Não é possível saber qual caso ocorrerá, pois depende das características dos vídeos e se o método automático conseguiu lidar com esses vídeos corretamente.

Figura 35 - Workflow da abordagem híbrida



Fonte: Elaborada pelo autor

A especificação dos pontos de sincronização candidato resultante da aplicação do método automático precisa ser avaliado e validado. A *crowd* entra no método híbrido proposto exatamente a partir desta etapa. Os *workers* recebem a tarefa para verificar se uma especificação de sincronização entre dois vídeos, gerada pelo método automático, está correto, impreciso ou errado. Uma especificação correta implica uma sincronização de vídeo acertada, que é então confirmada como o ponto de sincronização para esses vídeos. Uma especificação é imprecisa se a *crowd* considera que não é suficientemente precisa e deve ser melhorada. Ela será errada quando os *workers* considerarem que não há correlação entre os vídeos no ponto de sincronização marcado pelo método automático.

As especificações imprecisas são enviadas para serem refinadas (*Refiner* na Figura 35), sendo esta tarefa também realizada pela *crowd*. A tarefa de refinamento consiste em melhorar a precisão da sincronização gerada pelo processamento automático. Manualmente, o *worker* busca, segundo sua percepção, o melhor ponto de sincronização entre os vídeos (avançando cada vídeo *frame* a *frame*, reproduzindo e pausando o vídeo, movendo sua *timeline*, etc.). Essa nova especificação gerada pelo *worker* então é enviada mais uma vez para a etapa de Validação.

As especificações consideradas erradas podem ser enviadas para a etapa *CrowdSync* ou voltar para a fase automática na tentativa de sincronizar usando outro método automático. No experimento realizado, as especificações consideradas erradas pelos *workers* da etapa de validação são enviadas para outros *workers* do método, recrutados para a atividade *CrowdSync*. Nesta atividade, a tarefa dos *workers* é buscar pontos de sincronização nos vídeos assistindo seus conteúdos. Para encontrar esses pontos, os *workers* podem assistir a todo o conteúdo dos vídeos analisados, indicando um ponto a partir do qual seus conteúdos estariam sincronizados. Eles também podem assistir apenas a segmentos dos vídeos, reduzindo a complexidade da tarefa, mas elevando o número de passos necessários para completá-la.

Cada nova contribuição (valor de sincronização) gerada a partir da *crowd* é enviada para ser validada. Se um ponto de sincronização não puder ser encontrado entre um vídeo e o restante do conjunto já sincronizado, conclui-se que o vídeo não faz parte do conjunto solução final.

7.2.1. Detalhes de Implementação

A implementação da abordagem híbrida requereu o uso de um método automático de sincronização e de um sistema de *crowdsourcing* para gerenciar as fases destinadas à *crowd* no *workflow* da Figura 35.

A fase automática foi realizada utilizando o software *PluralEyes 4*, que analisa a trilha de áudio dos vídeos para sincronizá-los. O uso de um *software* comercial foi feito para garantir a qualidade da fase de sincronização automática. Por se tratar de um *software* fechado, não foi possível integrá-lo diretamente ao resto do sistema, no entanto, foi possível exportar a especificação de sincronização gerada pelo software através de um arquivo XML, permitindo o processamento da especificação e seu uso como entrada para as demais fases.

Para o uso da força de trabalho da *crowd*, foram desenvolvidas três aplicações para cada uma das fases em que os *workers* executam tarefas: validação, refinamento e sincronização. O desenvolvimento foi feito usando HTML5 para desenvolver as aplicações *frontend*, e no *backend*, NODE.JS. A *crowd* foi composta por 14 membros voluntários, pesquisadores do laboratório e alunos. Nas três aplicações, foram utilizadas as técnicas de consolidação baseada na função de

distribuição categórica das contribuições e o um limite superior de convergência (ver seção 5.5.1). Já a distribuição das tarefas, variou para cada aplicação.

A ferramenta de validação apresenta ao *worker* dois vídeos, que foram processados na fase automática do método. A saída desta fase é a especificação de um ponto de sincronização entre os conteúdos destes vídeos. A partir desta especificação, os vídeos podem ser apresentados lado a lado de forma sincronizada, conforme o exemplo da Figura 36.

Figura 36 - Ferramenta de Validação

Fonte: Elaborada pelo autor

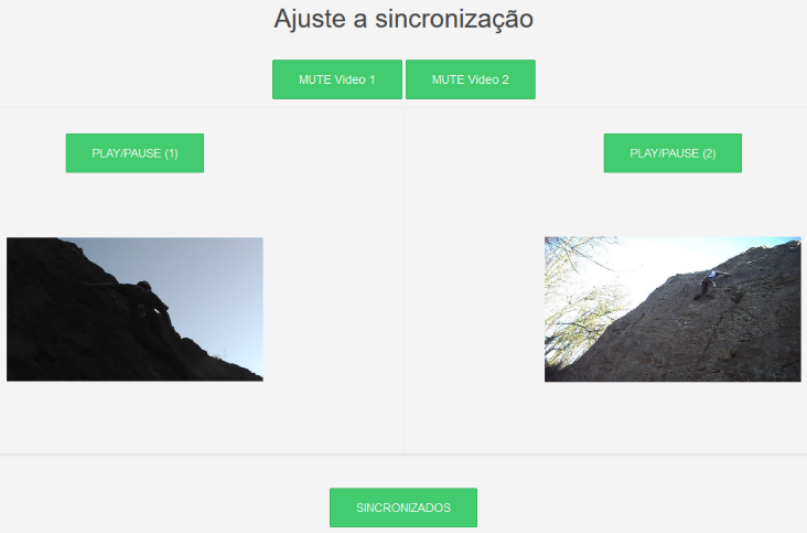
O *worker*, usando exclusivamente o seu ponto de vista, avalia se as apresentações dos dois vídeos estão sincronizadas. A partir disso, sua tarefa é indicar: (i) se ambos os vídeos estão perfeitamente sincronizados; (ii) se existe um problema menor, mas que o nível de sincronização obtido é aceitável; (iii) se existe um grande problema de sincronização, obrigando que ela seja refinada; ou ainda, (iv) se os vídeos não estão sincronizados porque não é possível identificar nenhum ponto de correlação entre os conteúdos do par de vídeos. Além de verificar a sincronização de forma visual, o *worker* pode interagir com os áudios, silenciando-os para ajudar no processo de identificar o *status* da sincronização.

A ferramenta de refinamento apresenta ao *worker* dois vídeos identificados pelos *workers* da fase anterior como imprecisos. A tarefa da *crowd* agora é melhorar a precisão da sincronização obtida pelo método automático.

A ferramenta de refinamento (Figura 37) junto com o par de vídeos define o *worker-space* nesse caso. Ele permite ao *worker* tocar e pausar cada um dos

vídeos, e navegar entre os seus *frames*, se necessário. Ao reproduzir e pausar o conteúdo dos vídeos, o *worker* realiza a tarefa de refinar a sincronização entre o par de vídeos.

Figura 37 - Ferramenta de Refinamento



Fonte: Elaborada pelo autor

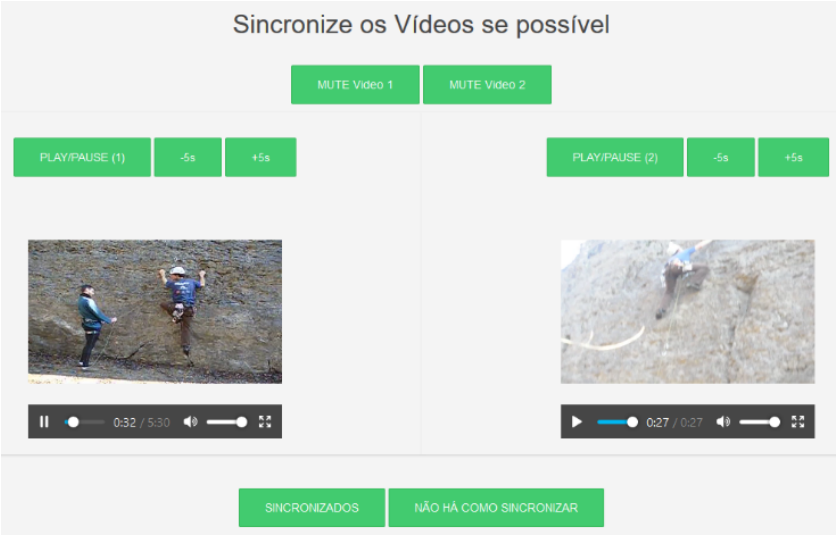
Na implementação realizada, a seleção dos vídeos enviados para os *workers* prioriza o seguinte aspecto: um dos vídeos do par enviado ao *worker* já possui um ponto de sincronização definido com outro vídeo; o outro vídeo é prioritariamente um pertencente ao subconjunto dos vídeos para os quais a especificação da sincronização foi classificada como errada na fase de validação.

Tentar priorizar a escolha de um dos vídeos, para o qual os pontos de sincronização foram considerados corretos, visa aumentar a probabilidade de correlação entre os conteúdos deste vídeo e o seu par, ainda não sincronizado e apresentado ao *worker*, conforme mostra a Figura 38. Um vídeo com maior número de pontos de sincronização já definidos com outros vídeos do conjunto de entrada, provavelmente é o que cobre a maior parte das ocorrências de interesse de um evento, talvez por ser o mais longo ou por não ter descontinuidade em seu conteúdo, ou ainda, por ter sido iniciado e concluído entre instantes de interesse do evento.

O *worker-space* para realização desta tarefa é composto pela interface da Figura 38 junto com os conteúdos completos do par de vídeos a ser analisado. Na interface, o *worker* dispõe de diversas funcionalidades para interagir com o conteúdo dos vídeos durante a execução da tarefa. Ele pode navegar pelos conteúdos usando barras de navegação, avançar ou atrasar os relógios de apresentação dos vídeos

usando os botões de controle (+5s e – 5s), e utilizar o conceito de *play & pause* para refinar a sincronização entre os vídeos.

Figura 38 - Ferramenta de Sincronização



Fonte: Elaborada pelo autor

7.2.2. Execução Experimento Híbrido

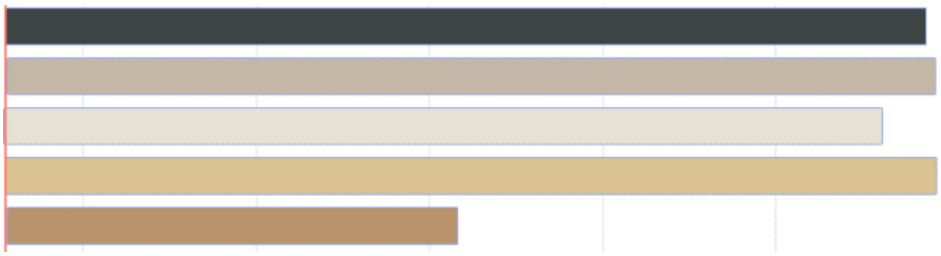
A abordagem híbrida foi aplicada sobre quatro conjuntos de vídeos, com o objetivo de avaliar se o uso da *crowd*:

- Acrescenta novos vídeos não capturados pelo método automático ao conjunto solução final esperado;
- Melhora a precisão da especificação dos valores de sincronização obtidos com o método automático.

Os quatro conjuntos de dados de entrada (*datasets*) contendo vídeos correlacionados utilizados no experimento envolvendo o método de sincronização híbrido foram os seguintes:

1. *Soccer* (SCHWEIGER, 2012): constituído por 5 vídeos de uma brincadeira de futebol. Os vídeos são filmados a partir de 5 diferentes câmeras fixas tentar cobrir o evento em 360º. Os cinco vídeos constituem um total de 1779s, em um intervalo de 406s (duração do evento). A Figura 39 mostra a distribuição dos vídeos no tempo. Nas figuras de representação do *Timeline*, dos datasets, cada cor representa uma câmera.

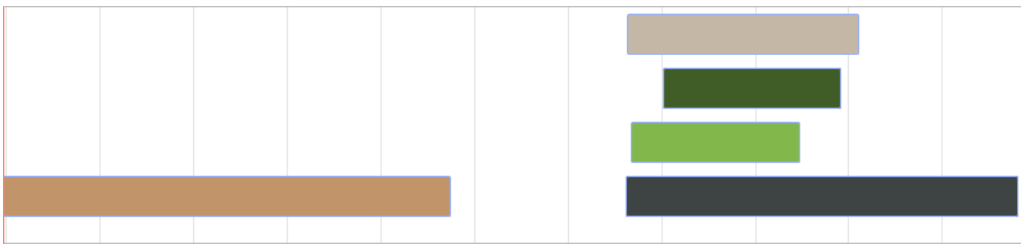
Figura 39 - *Timeline do dataset Soccer*



Fonte: Elaborada pelo autor

2. *Concert C2* (SHRESTHA, WITH, *et al.*, 2010): constituído por 5 vídeos de um show de rock, todos filmados por *smartphones* do público. As câmeras capturam o palco a partir da posição da plateia, mas com variações de ângulo devido ao diferente posicionamento das câmeras. O total de duração dos vídeos é de 1141,57s. Os vídeos cobrem duas partes distintas do show: 1 vídeo cobre o pré-show que dura 360,67s, enquanto que os outros 4 registram o show que tem duração de 315,7s. A Figura 40 mostra a distribuição dos vídeos no tempo.

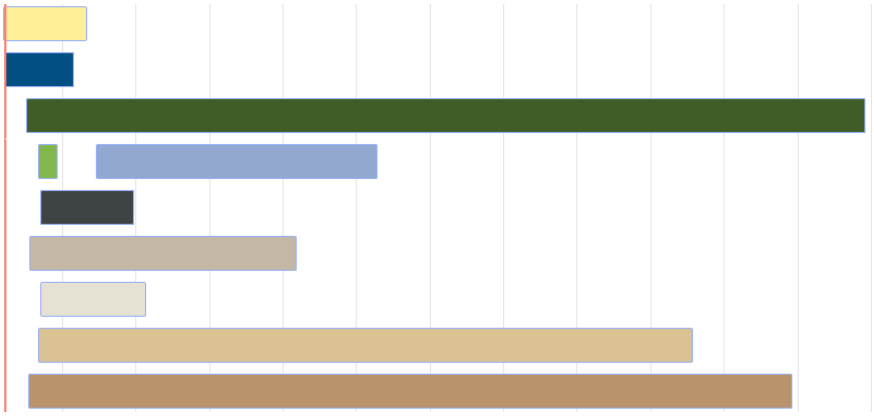
Figura 40 - Timeline do dataset C2



Fonte: Elaborada pelo autor

3. *Obama Speech*: uma coleção de 10 vídeos coletados do YouTube que mostram o discurso de vitória do presidente Barack Obama nas eleições de 2008 em Chicago. A duração total de todos os vídeos é de 6475s, em um intervalo de 1828s representando a duração do evento. A Figura 41 mostra a distribuição dos vídeos no tempo.

Figura 41 - Timeline do dataset Obama



Fonte: Elaborada pelo autor

4. *Climbing Dataset* (DOUZE, REVAUD, *et al.*, 2016): contém 89 vídeos de uma escalada, capturados por diferentes câmeras, incluindo telefones celulares, câmeras de filmagem e câmeras em capacetes. Este dataset foi apresentado no capítulo anterior e usado na simulação da participação da crowd para sua sincronização. A Figura 20 mostra a distribuição dos vídeos desta dataset no tempo.

7.2.3. Resultados do Método Híbrido

O método híbrido foi aplicado a cada um dos quatro *datasets* descritos na seção anterior. Alguns *datasets* foram sincronizados de imediato pelo método automático usado, restando à *crowd* apenas a tarefa de confirmar a sincronização. Em outros casos, alguns vídeos não foram sincronizados pelo método automático e tiveram que ser processados pelos *workers* a fim de melhorar o nível de sincronização.

7.2.3.1. Resultados com o dataset Soccer

A etapa de sincronização automática gerou uma especificação de sincronização contendo todos 5 vídeos presentes neste *dataset*. Essa especificação foi passada à fase de validação pela *crowd*.

Cada uma das 10 combinações possíveis de vídeos sobrepostos entre o conjunto de dados foi verificada por dois *workers*. Cada tarefa demorou cerca de 15s para ser executada. Como a *crowd* confirmou 100% da sincronização, não foi necessário refinar ou re-sincronizar qualquer vídeo. Com isso, o experimento foi encerrado.

Como forma de avaliar o grau de sincronização obtido pela ferramenta automática, o resultado obtido foi comparado ao apresentado por Schweiger (2013). No artigo o *offset* entre o vídeo da primeira e a quarta câmeras é de 4 *frames* (único valor informado), mesma precisão alcançada na especificação final gerada pelo método híbrido.

7.2.3.2. Resultados com o dataset Concert C2

A etapa de sincronização automática identificou pontos de sincronização entre quatro dos cinco vídeos do *dataset Concert C2*. O quinto vídeo não pode ser sincronizado com os demais na etapa automática.

Na etapa seguinte, a *crowd* avaliou apenas a precisão da sincronização entre os quatro vídeos encontrados. O quinto vídeo (câmera 5) foi enviado diretamente para a etapa de *CrowdSync*. Nessa etapa a *crowd* deveria tentar encontrar um ponto de sincronização entre o vídeo da câmera 5 com os das demais câmeras.

Na execução da validação, 12 contribuições foram necessárias para confirmar que a especificação gerada pela ferramenta automática estava correta. Devido a característica dos vídeos (*UGVs* e com pior qualidade), em média foram necessários 30s para execução das tarefas de validação.

Na etapa de sincronização pela *crowd*, o quinto vídeo foi comparado aos demais vídeos na tentativa de sincronizá-lo. Porém, nenhum ponto de sincronização foi encontrado ao final do processo, confirmando a saída do processo automático que definia aquele vídeo como independente dos demais.

Na verdade, o vídeo da câmera 5 foi filmado horas antes dos demais, impedindo qualquer sobreposição de conteúdo entre eles.

7.2.3.3. Resultados com o dataset Obama Speech

A etapa de sincronização automática produziu uma especificação sincronizando 9 dos 10 vídeos pertencentes ao *dataset Obama Speech*. Para um dos vídeos, o método não conseguiu revelar nenhum ponto de sincronização entre este vídeo e os demais.

Na etapa de validação, a especificação de sincronização para os 9 vídeos foi avaliada. Aqui pela primeira vez em todos os experimentos, a *crowd* identificou

problemas na sincronização gerada, marcando algumas relações como imprecisas. Das 19 relações de sincronização entre os 9 vídeos encontradas pelo método automático, 7 foram apontadas como imprecisas pela *crowd*, indicando a necessidade de uma fase de refinamento. Para a *crowd* identificar estas 7 relações imprecisas e validar as outras 12 como corretas, foram necessárias 57 contribuições, com uma duração média de 19s para a realização de cada tarefa.

Na fase de refinamento, cada uma das 7 relações apontadas como imprecisas foi enviada a um *worker*. Cada relação foi analisada por dois *workers*, gerando 14 contribuições nesta etapa. Após execução do refinamento, a melhoria das 7 relações foi, em média, de 0,474s, com a redução do erro médio de 1,008s para 0,534s.

Em paralelo a etapa de refinamento, o *CrowdSync* foi executado para tentar sincronizar o único vídeo não sincronizado aos outros pelo método automático. Ao contrário do ocorrido com o *dataset Concert C2*, neste experimento o vídeo foi sincronizado aos outros com sucesso pela *crowd*. Mais ainda, bastaram duas contribuições para que um ponto de sincronização deste vídeo com o conjunto fosse encontrado.

Este vídeo não foi sincronizado como os demais vídeos pela etapa automática porque sua trilha de áudio continha interferências causadas por comentários em japonês sobre o discurso de Obama ao longo de sua duração. Esses comentários interferiram diretamente no método de sincronização baseado nas trilhas de áudio dos vídeos analisados.

7.2.3.4. Resultados com o dataset Climbing

Entre os 4 conjuntos de dados, o *climbing* representa o mais heterogêneo em termos de conteúdo, possui o maior número de vídeos a serem sincronizados, além de apresentar problemas de oclusão e envolver captura com câmeras estáticas e móveis a diferentes distâncias do ponto de interesse principal do vídeo.

Após a execução do método automático, 76 dos 89 vídeos obtiveram uma especificação de pontos de sincronização entre seus conteúdos, enquanto para 13 deles, não foi possível identificar nenhum ponto de sincronização com os demais vídeos.

A partir dos 76 vídeos resultantes do método automático, podem ser construídas 394 relações de sincronização. Cada uma destas relações foi validada por dois *workers*, requerendo assim 788 contribuições para a conclusão desta etapa do processo híbrido executado pela *crowd*. A *crowd* classificou 29 das 394 relações de sincronização obtidas de maneira automática como imprecisas. Uma análise mais precisa do resultado demonstrou que todas as 29 relações julgadas imprecisas se relacionavam a um subconjunto de 6 vídeos obtido do conjunto inicial gerado pelo método automático, contendo 76 vídeos. Este subconjunto de 6 vídeos foi então enviado para a etapa de refinamento realizada pelos *workers*.

Antes da fase do refinamento, o erro médio de sincronização para os 76 vídeos em relação ao *Ground Truth* do *dataset* era de 1,15s, com desvio padrão (σ) igual a 0,50. Já o erro médio do subconjunto de 6 vídeos considerados imprecisos na fase de validação foi de 1,28s (σ=0,54). Após 8 contribuições dos *workers* na tarefa de refinamento, o erro médio nas relações dos 6 vídeos imprecisos caiu para 0,35s (σ =0,41) e o erro geral foi reduzido para 1,10s (σ =0,59).

Na sincronização realizada pela *crowd*, os 13 vídeos que não puderam ser sincronizados pelo método automático foram enviados aos *workers*. A tarefa a ser realizada por cada *worker* consistia em buscar pontos de sincronização destes vídeos com os outros 76 já sincronizados. Ao final da etapa, 12 dos 13 vídeos puderem ser sincronizados com os demais, enquanto que para 1 deles não foi possível encontrar qualquer ponto de sincronização com os vídeos restantes. Para que estes novos pontos de sincronização entre o subconjunto de 12 vídeos e os outros 76 gerados na pelo método automático foram necessárias 228 contribuições dos *workers*. Para os novos 12 pontos de sincronização gerados, o erro médio foi de 1,28s (σ=0,47).

7.2.3.5. Sumarização do Experimento Híbrido

Os resultados obtidos para cada um dos *datasets* testado foram bastante diferentes dos demais, principalmente devido às características variadas dos vídeos que os compõem. O *dataset Soccer*, bastante homogêneo, obteve ótimo resultado com o método automático de sincronização. O mesmo comportamento foi observado para o *dataset Concert C2*. A única diferença no processamento automático destes dois *datasets* foi que 1 dos vídeos do *Concert C2* não pode ser sincronizado aos

outros 4 restantes daquele conjunto. Isso obrigou que contribuições adicionais da *crowd* fossem realizadas para confirmar que esse vídeo não possuía, realmente, nenhum ponto de sincronização com os demais.

O *dataset* Obama foi o primeiro dos *datasets* utilizados no experimento híbrido a ter valores de sincronização imprecisos após a aplicação do método automático de sincronização. Em outras palavras, este *dataset* não conseguiu ter sucesso na sincronização de todos os seus vídeos com a aplicação de um método automático. Como foi mencionado na seção 7.2.3.3, um dos vídeos do *dataset* possuía uma trilha de áudio com comentários em japonês durante o discurso, causando uma interferência no áudio e impedindo sua sincronização. Conforme também relatado na mesma seção, a etapa de sincronização pela *crowd* obteve sucesso na sincronização deste vídeo com os demais.

O último *dataset* testado (*Climbing*) acabou confirmando a suposição inicial de que seria o mais difícil a ser sincronizado. A etapa automática permitiu sincronizar com sucesso mais de 70% dos vídeos. Por outro lado, vários dos pontos de sincronização encontrados estavam imprecisos, além de não ter sido possível encontrar pontos de sincronização para diversos vídeos do *dataset*. Em ambos os casos, a *crowd* teve que ser utilizada para refinar ou melhorar os resultados.

No caso da etapa de refinamento, os *workers* analisaram 6 vídeos com sincronização identificada como imprecisa e, depois da análise, alteraram os instantes de sincronização obtidos pelo método automático. A precisão global obtida para as sincronizações entre os vídeos não foi grande, mesmo com o uso da *crowd*. Isso ocorreu por que alguns atrasos entre os conteúdos dos vídeos não foram percebidos pelos *workers*.

No caso da etapa de *CrowdSync*, apesar dos *workers* terem sucesso na sincronização dos vídeos que não puderam ser sincronizados pelo método automático, verificou-se que o erro final obtido foi relativamente alto, superior a 1s em média. Tal fato reforça a ideia de que tanto a computação humana, quanto a computação por máquina podem gerar falhas no processamento de entradas muito complexas e pouco homogêneas.

A Tabela 3 sumariza os resultados obtidos nos experimentos híbridos envolvendo uma fase de processamento automático seguida de uma fase de processamento usando *crowdsourcing*.

Tabela 3 - Etapas da execução do método híbrido

Dataset	Automática[1]	Validação[2]	Refinamento[3]	Crowdsync[4]	Resultado[5]
Soccer	5/5 0s	0/5 20	0/0 0	0/0 0	5/5 0s
C2	4/5 0,12s	4/4 12	0/0 0	0/1 2	4/5 0,12s
Obama	9/10 0,9s	6/9 57	3/3 14	1/1 2	10/10 0,58s
Climbing	76/89 1,15s	70/76 788	6/6 16	12/13 228	88/89 1,10s

1) N° vídeos corretamente sincronizados / N° de vídeos analisados e erro médio da sincronização;
2) N° de vídeos identificados como precisos / N° de vídeos analisados e N° de contribuições necessárias;
3) Erro após refinamento / Erro antes do Refinamento e número de contribuições;
4) N° de vídeos corretamente sincronizados/ N° de vídeos analisados e erro médio da sincronização;
5) N° de vídeos corretamente sincronizados/ N° de vídeos e erro da precisão.

7.3. EXPERIMENTO FINAL: CROWDSYNC METHOD IN THE WILD

O recrutamento da *crowd*, validação de suas contribuições e agregação destas em uma solução final são alguns dos maiores desafios para os sistemas de *crowdsourcing*. Durante a descrição da pesquisa e apresentação dos demais experimentos, as questões de agregação e validação foram diretamente abordadas.

A agregação pode ser realizada com o uso de técnicas baseadas em convergência e inferência (ver seção 5.5). Já a validação pode usar técnicas baseadas na *gamificação* ou, no caso específico da sincronização de vídeos, usar a distribuição categórica para separar as contribuições aleatórias e discrepantes das contribuições reais.

No entanto, até este ponto do trabalho, muito pouco foi explorado sobre o recrutamento da *crowd*. Apesar da descrição do *CrowdVideo* da seção 5.3 mencionar a questão do perfil dos *workers* e das recompensas pelas tarefas, os experimentos tratados nas seções 7.1 e 7.2 utilizaram apenas *crowds* controladas formadas por voluntários, em geral, pessoas próximas ao autor desta tese. Sendo assim, para que esta pesquisa estivesse completa, foi necessário o recrutamento de uma *crowd* real, na qual não houvesse nenhum tipo de controle sobre os

participantes, mas com a implementação do controle no processo de envio de tarefas e agregação dos resultados produzidos individualmente por cada *worker*.

O princípio fundamental do método *crowdsourcing* é que *workers* quaisquer sejam recrutados para executarem tarefas individuais (simples) produzindo resultados que, combinados, gerem uma solução final para determinado problema. No caso desta tese, o problema a ser solucionado era o da sincronização entre *UGVs* correlacionados a um mesmo evento social.

O “Experimento Final” foi realizado com o objetivo de que uma *crowd*, suficientemente genérica, fosse recrutada para resolver o problema da sincronização de um conjunto de vídeos correlacionados. O que se deseja avaliar é se uma *crowd* não voluntária, sem perfil definido e distribuída pela Internet poderia realizar a *task* de encontrar pontos de sincronização e gerar uma especificação de sincronização dos vídeos.

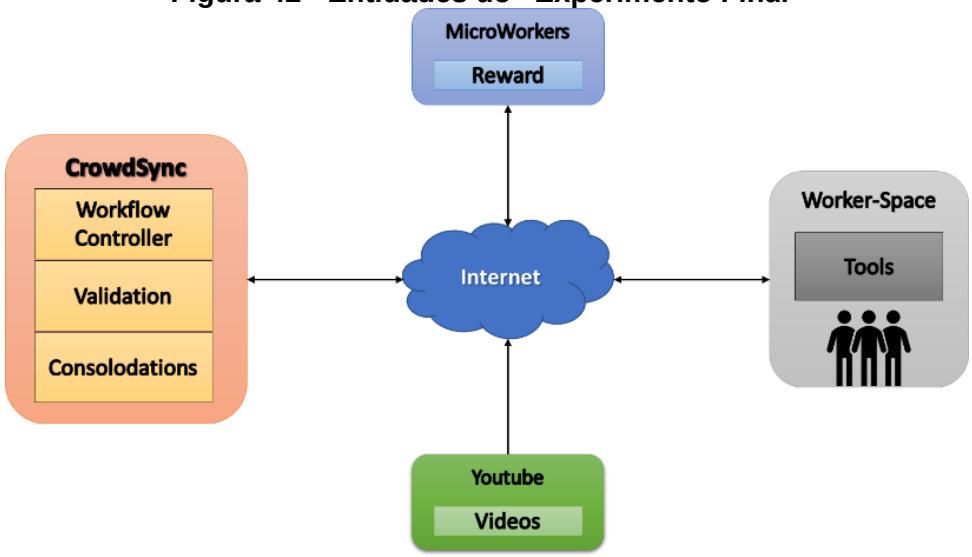
A principal diferença deste experimento para os anteriores é o uso de uma *crowd* recrutada com o uso de uma plataforma comercial de *crowdsourcing*. Nesta plataforma, os *owners* podem criar *tasks* a serem realizadas e, a partir da oferta de pagamentos, convocar *workers* que irão executar as *tasks*.

A plataforma comercial escolhida para o experimento foi a *MicroWorkers* (*microworkers.com*). Ela possui diversos usuários cadastrados e disponíveis para realizar as tarefas propostas pelos *owners* em troca de recompensa monetária. Esta plataforma foi escolhida por permitir a integração de suas funcionalidades (como o pagamento e recrutamento de *workers*) com aplicações externas. Assim, foi possível manter todas as funcionalidades originais do sistema de sincronização desenvolvido no escopo desta tese, incluindo apenas modificações na forma de recrutamento, recompensa e disponibilização do *worker-space*.

A Figura 42 mostra como foi feita a integração da aplicação de sincronização com a plataforma *MicroWorkers*. Uma aplicação é desenvolvida de forma a ser compatível com a plataforma e esta aplicação passa a ser a ferramenta usada pelo *worker* para executar sua tarefa. A aplicação deve ser hospedada dentro da plataforma *MicroWorkers* e ser capaz de comunicar-se com o servidor que hospeda as demais funcionalidades do *CrowdSync*, tais como o controlador de tarefas, a validação das contribuições e agregação do resultado. Quando pronta, a proposta

de trabalho é oferecida aos *workers*, juntamente com a descrição da tarefa, aplicação e o pagamento por sua execução. Após aceitar a proposta de trabalho no *MicroWorkers*, um *worker* acessa a aplicação a partir do seu próprio *browser*. Durante o acesso, a aplicação se comunica diretamente com o servidor *CrowdSync*, que envia ao *worker* o par de vídeos a ser sincronizado. Imediatamente, após receber a identificação dos pares de vídeos, a aplicação se conecta ao *YouTube*, que serve de repositório para os vídeos para o experimento. Após carregar o par de vídeos, o *worker* pode executar a tarefa de encontrar um ponto de sincronização entre o par de vídeos. Após concluir a tarefa, o *worker* envia sua contribuição ao servidor *CrowdSync* e também uma confirmação para que o *MicroWorkers* libere seu pagamento.

Figura 42 - Entidades do "Experimento Final"

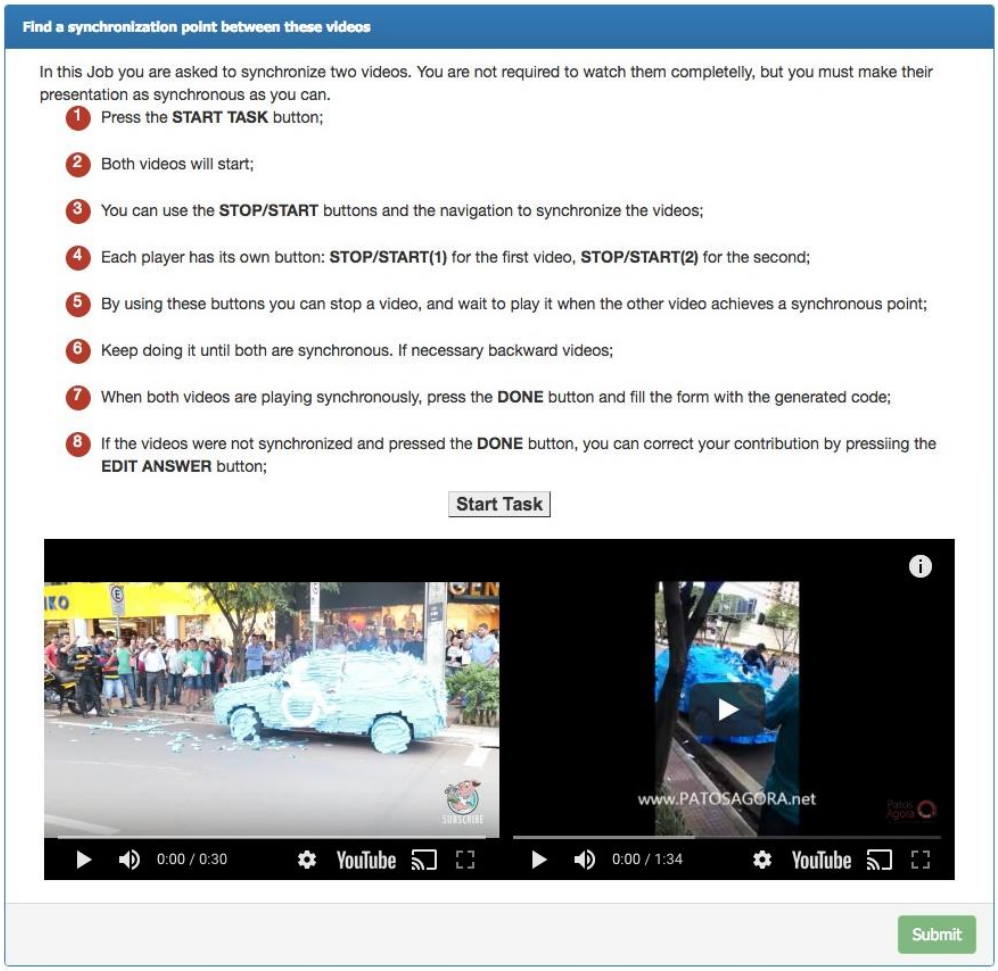


Fonte: Elaborada pelo autor

Para a construção da ferramenta foram utilizadas como base as aplicações de sincronização já construídas nos experimentos anteriores, pois o seu desenvolvimento havia sido realizado em HTML5 e a plataforma *MicroWorkers* suporta a utilização de aplicações HTML5 para personalização das aplicações, sendo necessário apenas a adaptação para a comunicação com as outras entidades, como questões de segurança *Cross-Domain.*

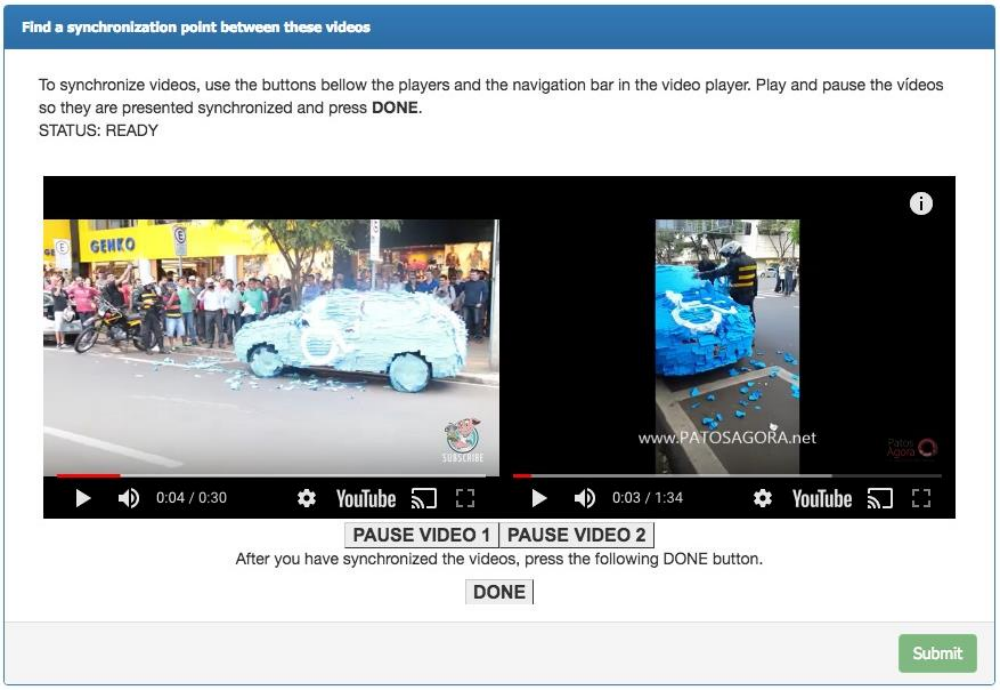
A Figura 43 ilustra as instruções enviadas a um *worker* para que ele complete sua *task*. Após a etapa de leitura, ao apertar o botão de inicialização da tarefa na mesma figura, a interface é alterada, passando a apresentar a reprodução lado a lado do par de vídeos a ser analisado e dois botões do tipo *Play & Pause* abaixo de cada vídeo, como mostra a Figura 44.

Figura 43 - Instruções para realização da *Task*



Fonte: Elaborada pelo autor

Figura 44 - Interface de Sincronização da Task



Fonte: Elaborada pelo autor

Com o uso dos botões *Play & Pause* e/ou da barra de navegação de cada vídeo, o *worker* pode adiantar, pausar, tocar e atrasar seus conteúdos até chegar a um ponto no qual ele assume que os vídeos estão sincronizados. Para concluir a tarefa, o *worker* aperta o botão *DONE*, que sinaliza ao *CrowdSync* que a tarefa foi finalizada e habilita a opção de *SUBMIT*, que é um botão do *MicroWorkers* para encerrar a tarefa e sair da tela de aplicação.

O uso do método de sincronização baseado nas ações *Play* & *Pause* sobre os vídeos permitiu que os *workers* executassem uma atividade muito simples. Por outro lado, foi observado que o método tem problemas quando se deseja maior precisão na sincronização, uma vez que os ajustes são feitos enquanto o vídeo é reproduzido e sem possibilidade de um ajuste mais fino, como no caso da navegação *frame* a *frame*, por exemplo.

A distribuição das tarefas de sincronização para os *workers* foi realizada de duas formas no “Experimento Final”. Primeiro, houve uma distribuição uniforme, onde cada par de vídeos recebeu o mesmo número de contribuições. Em seguida, o método da lista de distribuição foi aplicado, além da forma de definição do número de contribuições requisitadas ter sido alterada.

Como o *MicroWorkers* requer que o número de contribuições solicitadas aos *workers* seja definido, ele foi fixado em 30 para o experimento. Isso quer dizer que mesmo que todas relações de sincronização definidas pelos *workers* convergissem com um número de contribuições inferior a 30, ainda assim outras contribuições continuariam a ser solicitadas até que esse número fosse alcançado.

O experimento, porém, foi dividido em três campanhas, que é o termo utilizado pela plataforma *MicroWorkers*. Cada campanha teve um *dataset* diferente como entrada, além de alguns objetivos específicos. Mais ainda, em cada campanha foram medidos a precisão das contribuições (valor do *offset* entre os dois vídeos da contribuição), o tempo para realizar a tarefa e a quantidade de interações com os botões de *Play* & *Pause.* Informações como país de origem das contribuições e tempo para leitura das instruções também foram coletadas nos experimentos.

7.3.1. Primeira Campanha

A primeira campanha teve como objetivos principais:

- Avaliar o correto funcionamento da integração entre a ferramenta de sincronização e o recrutamento da *crowd* via o *MicroWorkers*;
- Avaliar se os *workers* conseguiriam realizar as tarefas a partir das instruções e funcionalidades da ferramenta de sincronização.

Ela utilizou 3 vídeos como conjunto de entrada, os quais geram 3 combinações de relações temporais possíveis entre eles. Como um dos focos desses testes é averiguar se os *workers* são capazes de realizar a sincronização, não importa quantas relações são avaliadas, mas sim ter várias contribuições executadas.

Outro fator a ser destacado nesta campanha é o conjunto de vídeos de entrada escolhido. Como forma de remover variáveis que poderiam dificultar a execução com sucesso das *tasks*, criou-se um conjunto de entrada formado por diversos segmentos de vídeos de duração variável extraídos de um mesmo vídeo original. A ideia foi simular o uso de múltiplas câmeras para capturar um conteúdo, tendo a certeza de que sempre seria possível encontrar a superposição entre as partes. Uma outra vantagem dessa escolha é que a tarefa do *worker* de inspecionar visualmente o conteúdo dos vídeos em busca de um ponto de sincronização é facilitada, pois os pares de vídeos analisados, por serem originados de um mesmo vídeo, terão conteúdos exatamente iguais sendo apresentados após a sincronização. Os segmentos de vídeos utilizados foram gerados a partir do vídeo SINTEL (https://durian.blender.org/).

Um total de 30 contribuições foram realizadas pelos *workers* na 1ª campanha, a um custo total de US$ 1,80, sendo US$ 0,30 de taxa de uso do *MicroWorkers* e US$ 0,05 por cada contribuição.

Três tempos diferentes foram medidos nesta 1ª campanha: (i) o tempo total de execução de uma tarefa; (ii) o tempo de execução da etapa de sincronização e (iii) o tempo gasto na leitura das instruções. O tempo levado para realização da tarefa como um todo foi em média de 207,1s (σ =127,615).

No geral, os 3 valores de tempos anteriores tiveram grande variação. Por exemplo, alguns *workers* terminaram as tarefas em menos de 1 min, enquanto outros precisaram de mais de 9 min para sincronizar o par de vídeos recebido. Ao final, o tempo de execução da etapa de sincronização para a 1ª campanha (sem o tempo para leitura das instruções) foi em média 83,88s (σ =67,52). Com relação à

precisão atingida, apenas 2 das 3 das relações possíveis entres os vídeos foram obtidas pelos *workers*.

Um erro na comunicação entre a aplicação e o controlador de tarefas durante a execução da 1ª campanha acabou prejudicando os resultados obtidos. O erro fez com que as tarefas não fossem corretamente distribuídas entre os *workers*, o que fez com que os números de contribuições da *crowd* para as relações[2] fossem iguais: R(1-2)=19; R(2-3)=10 e a R(1-3)=1. Assim, não foi possível avaliar a precisão dos valores obtidos para a relação R(1-3), que foi definida em cima da contribuição de apenas 1 *worker*. Os valores obtidos para as duas relações, R(1-2) e R(2-3), foram avaliadas de forma individual.

A Tabela 4 sumariza os resultados alcançados na 1ª campanha. Além do número de contribuições associadas a cada relação, a tabela apresenta: o Δ esperado, que é o valor encontrado por um especialista para a sincronização dos vídeos; o Δ médio, encontrado a partir da média de todas as contribuições para aquela relação e; o Δ categórico, obtido com a aplicação do algoritmo de categorização aos valores das contribuições em um intervalo de 2s, exigindo uma confiança maior que 0,5 para convergir para um valor de sincronização.

Tabela 4 - Precisão da Campanha 01 (valores absolutos)

	N. Contr.	*Δ Esperado*	*Δ Médio*	*Δ Categórico(2s, 0.5)*
R(1-2)	19	59,22	58,07 (σ =6,13)	59,11 (σ =0,42)
R(1-3)	1	96,15	10,26	**INSUFICIENTE**
R(2-3)	10	NULL	20,90 (σ =17,66)	**DIVERGÊNCIA**

Como pode ser visto na Tabela 4, a R(1-2) alcançou um valor médio de 58,07s, cerca de 1s de erro com relação ao valor Δ esperado. Após a categorização, mesmo em um intervalo amplo de 2s, o valor de sincronização para a mesma relação passa para 59,11s.

A R(1-3) obteve apenas uma contribuição, que por sua vez resultou em um valor errado e longe do valor de sincronização esperado. Não é possível verificar a convergência desta relação por motivos óbvios.

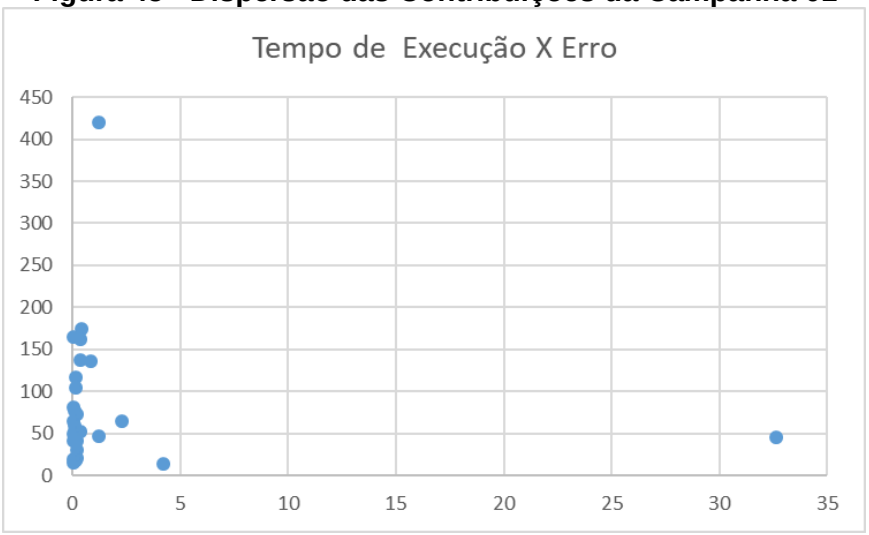
A R(2-3) se difere das demais por que se sabia de antemão que não seria possível encontrar uma relação de sincronização entre os vídeos v_2 e v_3. Para analisar o comportamento da *crowd* neste tipo de cenário, propositalmente não foi

[2] As relações de sincronização entre dois vídeos v_1 e v_2 será denotada por R(1-2) para todos os experimentos realizados nesta tese.

definida uma opção para indicar que não há sincronização entre o par de vídeos na ferramenta desenvolvida. Com isso, os *workers* foram obrigados a escolher um valor arbitrário de sincronização. Isso gerou uma grande variação dos valores das contribuições, os quais, quando categorizados, revelaram uma situação de divergência, confirmando a não existência de um ponto de sincronização entre os vídeos v_2 e v_3.

Os *workers* tinham a opção de executar a tarefa de sincronizar os vídeos tanto por meio de interações sobre os botões de *Play* & *Pause* da interface, quanto pela barra de navegação dos *players*. O comportamento esperado foi o observado. Houve uma distribuição uniforme entre o uso dos botões e da barra. Para as 19 contribuições da R(1-2), 12 foram obtidas a partir dos botões e 7 com o uso da barra de navegação. Por outro lado, ainda com relação às interações do usuário com a interface na 1ª campanha, um resultado interessante e inesperado foi observado. O uso da barra gerou contribuições com valores muito semelhantes àquelas produzidas a partir da interação com os botões. A melhor precisão alcançada utilizando botões foi de 59,17s, enquanto que com a barra de navegação, o melhor resultado obtido foi de 59,18s.

Figura 45 - Dispersão das Contribuições da Campanha 01



Fonte: Elaborada pelo autor

A Figura 45 apresenta a dispersão das contribuições para a R(1-2) de acordo com o tempo de execução da *task* e com o erro do valor de sincronização. É possível notar que (i) os pontos de sincronização foram identificados rapidamente, (ii) que um tempo maior na execução da tarefa não implica diretamente uma melhora no nível de sincronização e, (iii) que contribuições mais rápidas também não

garantem uma contribuição mais precisa. Entretanto, foi possível inferir que *workers* mais capacitados são capazes de encontrar o ponto de sincronização de forma rápida, justificando a concentração de pontos com um erro próximo a 0, no intervalo de 0-100s na duração da *task*.

7.3.2. Segunda Campanha

Com o término e análise da 1ª campanha, foi iniciada uma nova campanha com o objetivo de corrigir os erros de distribuição de tarefas apresentados e de modificar as características do conjunto de vídeos utilizados no experimento. Ao invés de utilizar segmentos extraídos de um mesmo vídeo, desta vez foram utilizados os vídeos do *dataset Obama*, discutido na seção 7.2.3.3. Dos 10 vídeos do *dataset* original, 4 foram selecionados para a campanha de sincronização. A seleção incluiu 3 vídeos com conteúdos visual bastante diferentes, sendo um deles filmado a partir do público e outro, o vídeo que havia provocado problemas para a aplicação do método automático, conforme discutido na seção citada. Relembrando, o método automático não conseguiu encontrar o ponto de sincronização esperado para o vídeo mencionado por causa dos trechos com tradução do discurso para o idioma Japonês. A pretensão inicial seria a de utilizar o mesmo conjunto de 10 vídeos do *dataset Obama*. Porém, recentemente, durante o desenvolvimento desta tese, 4 dos 10 vídeos originais foram removidos do YouTube. Assim, apenas 4 dos vídeos pertencentes ao *dataset* usado anteriormente foram usados nesta 2ª campanha, cujos objetivos são:

- Verificar a correta distribuição dos pares de vídeos;
- Avaliar a sincronização realizada com o uso de vídeos heterogêneos;

O custo para esta 2ª campanha é exatamente o mesmo da campanha anterior, isto é, US$ 0,30 de taxa de uso do *MicroWorkers* adicionado ao US$ 1,50 referente às 30 contribuições individuais com custo igual a US$ 0,05, perfazendo um total de US$ 1,80.

O tempo médio total medido entre o instante que o *worker* recebe a tarefa e concluiu a sua execução na 2ª campanha foi de 254s (σ =120,29). Este valor foi superior ao da 1ª campanha. A média de execução da sincronização em si, desconsiderando o tempo de leitura das instruções, foi de 119,45s (σ =105,14). O

aumento do tempo para a execução das tarefas pode ser justificado pela maior duração dos vídeos (média de 620s nesta campanha contra 68s na primeira) com relação àquelas dos vídeos do *dataset* anterior.

É possível definir 6 relações de sincronização para os 4 vídeos no *dataset Obama*. Ao contrário da 1ª campanha, não foi verificado nenhum problema com a distribuição das tarefas para os *workers* recrutados com a *MicroWorkers*. Com isso, cada uma das 6 relações recebeu 5 contribuições da *crowd*. A Tabela 5 sumariza os resultados obtidos na 2ª campanha.

Tabela 5 - Precisão da Campanha 02 (valores absolutos)

	N. Contr.	*Δ Esperado*	*Δ Médio*	*Δ Categórico (2s, 0.5)*
R(1-2)	5	93,27	96,10 (σ =5,78)	93,21 (σ =0,07)
R(1-3)	5	90,31	90,53 (σ =0,61)	90,53 (σ =0,61)
R(1-4)	5	27,77	53,33 (σ =53,15)	26,76 (σ =0,31)
R(2-3)	5	2,72	1,46 (σ =1,27)	**INSUFICIENTE**
R(2-4)	5	120,1	98,17 (σ =44,78)	120,36 (σ =0,39)
R(3-4)	5	117,22	117,40 (σ =0,23)	117,40 (σ =0,23)

Após análise das contribuições da 2ª campanha, foi possível verificar o bom nível de sincronização obtido pela *crowd*. Com respeito à média simples das contribuições, o valor da R(1-2) ficou apenas alguns segundos distante do valor esperado de sincronização. Com a aplicação da categorização, o valor final de erro caiu para 0,06s. O mesmo comportamento vale para as relações R(2-4) e R(1-4). Já as relações R(1-3) e R(3,4) tiveram valores muito próximos para a média pura e a categórica, já que todas as contribuições dos *workers* para estas duas relações produziram valores que se encaixaram no intervalo de categorização de 2s.

Diferente das demais relações, R(2-3) não obteve contribuições suficientes para que fosse possível definir o intervalo de categorização predominante. As cinco contribuições para R(2-3) geraram os seguintes valores: 0,12s; 2,54s, 0,38s, 2,81s e 12,46s. Estes cinco valores geram 3 diferentes categorias, cada uma delas com probabilidade inferior a 50% de ocorrência. Sendo assim, não foi possível identificar qual categoria predomina e não ocorre a convergência da relação.

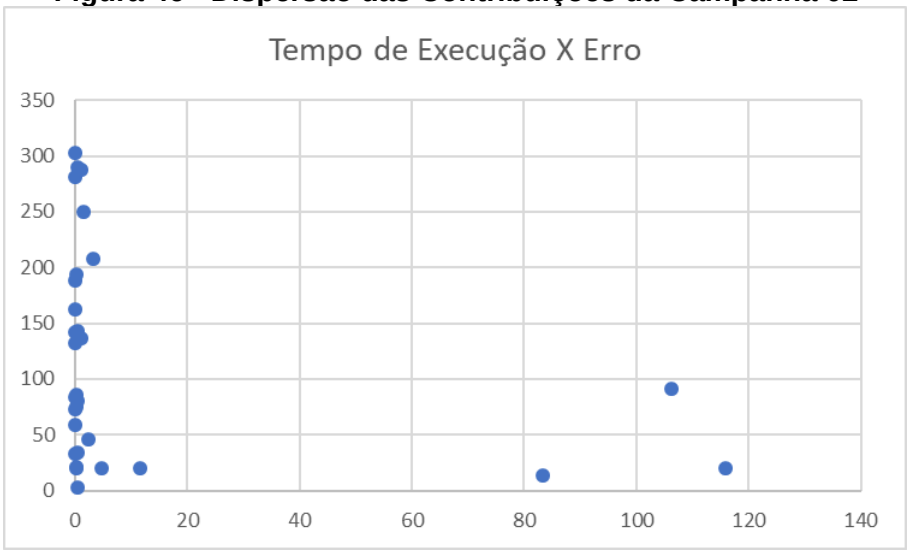
A variação de valores obtidos para a relação R(2-3) se deve às características dos vídeos envolvidos na tarefa de sincronização. O vídeo v_2 é proveniente de uma câmera de um usuário no meio da multidão assistindo ao discurso de *Obama*, porém a filmagem não consegue mostrar o palco onde o ex-presidente fazia seu discurso e, com isso, o *worker* tinha que se guiar apenas pelo áudio para identificar os pontos

de sincronização entre os vídeos envolvidos. Já o vídeo v_3 corresponde a uma filmagem do palco e das ações do presidente e possui uma trilha de áudio com a tradução em tempo real para japonês do discurso de *Obama*. Dessa forma, o nível de dificuldade da sincronização deste par de vídeos é maior. Apesar desta dificuldade, os valores de contribuição 2,54s e 2,81s mostram que dois *workers* conseguiram identificar o ponto de sincronização esperado, o que não foi possível para o método automático utilizado no experimento da 7.2.3.3. Os valores 0,12s e 0,38s indicam o fato de que os *workers* que executaram a tarefa não interagiram com a interface de apresentação dos vídeos em nenhum instante. De fato, eles apenas iniciaram a reprodução dos vídeos, mas provavelmente por não encontrarem um ponto de sincronização entre eles, simplesmente apertaram o botão DONE (ver Figura 44) para encerrar a tarefa. Outra observação é que estes valores muito baixos deveriam ter sido computados como 0s, mas isso não foi possível devido a uma limitação na inicialização dos *players* de vídeo. Essa limitação insere um pequeno valor de *offset* entre o início da reprodução de cada vídeo, sendo esse atraso independente de qualquer interação do *worker*.

Com relação ao uso da interface do *worker-space*, 18 *workers* utilizaram exclusivamente a barra de navegação do *player* para executar a tarefa de sincronização, enquanto que 12 utilizaram apenas o par de botões de *Play* & *Pause*. Mais ainda, os resultados obtidos com o uso de ambas as formas de interação atingiram, assim como na 1ª campanha, um nível de precisão semelhante para a sincronização do par de vídeos.

A Figura 46 mostra a dispersão das contribuições para a 2ª campanha, de acordo com o tempo gasto na execução da tarefa e o erro de cada uma com respeito ao *Golden Standard*.

Figura 46 - Dispersão das Contribuições da Campanha 02



Fonte: Elaborada pelo autor

7.3.3. Terceira Campanha

A 3ª campanha foi criada com o intuito de realizar um teste com um número maior de relações, em um *dataset* totalmente composto por *UGVs* e coletando um número maior de contribuições do que nas campanhas anteriores. Os 8 vídeos pertencentes a este *dataset* foram capturados por pessoas na rua, que filmaram um evento social comum. No evento, as pessoas capturaram a reação de um motorista ao ter seu carro coberto por *post-its* como forma de "punição coletiva" por ter estacionado seu veículo em uma vaga destinada exclusivamente a deficientes. Todos os vídeos foram resultados de capturas independentes e voluntárias realizadas por câmeras pessoais, mostrando diferentes ângulos do evento de interesse. Em especial, o processamento de 3 dos 8 vídeos impõe um desafio a mais aos *workers*. Eles possuem uma descontinuidade em seu conteúdo devido às pausas durante as suas capturas. Mais precisamente, os vídeos 5, 6 e 7 possuem cortes de cena de 9,63s, 9,63s e 42,75s de duração, respectivamente. Além disso, os vídeos 5 e 6 correspondem a variações da mesma filmagem, mas com zoom e qualidade diferentes.

O uso do subconjunto de 3 vídeos anteriores tem como objetivo descobrir quais tipos de contribuições serão geradas e, se mesmo diante de maiores dificuldades, a *crowd* ainda é capaz de encontrar pelo menos um dos pontos de sincronização entre esses vídeos.

O *dataset* de vídeos é composto por 8 vídeos com cerca de 84s cada, permitindo que 28 relações temporais sejam estabelecidas entre eles. Para sincronizar os vídeos do *dataset*, foi previsto um total de 113 contribuições aos *workers* recrutados pela *Microworkers*. O custo total da 3ª campanha foi de US$ 5,95, sendo calculado pela soma dos US$ 0,30 de taxa de uso do *MicroWorkers* com os U$ 5,65 referentes às 113 contribuições dos *workers*, a um custo de US$ 0,05 por tarefa concluída.

O número de contribuições para cada relação foi variável, já que a gestão de contribuições para esta campanha usou a distribuição categórica para determinar se os valores das relações já haviam convergido. O uso da distribuição categórica acabou mostrando que, na prática, a previsão inicial de que 113 contribuições fossem suficientes para se obter a sincronização de todos os vídeos do *dataset* e mostrou equivocada.

O tempo total para execução das task pelos *workers* foi em média 212,57s (σ=124,37). Desconsiderando o tempo para a leitura das instruções, a parte de sincronização exigiu 81,24s (σ=71,37) de dedicação dos *workers*, valores próximos aos obtidos na 1ª campanha.

A Tabela 6 sumariza os resultados da 3ª campanha, destacando os valores das contribuições associadas às combinações possíveis entre os vídeos v_1, v_3, v_4 e v_8, que possuem como característica comum, a não descontinuidade de seus conteúdos.

Tabela 6 - Precisão da Campanha 03 (valores absolutos) - A

	N. Contr.	*Δ Esperado*	*Δ Médio*	*Δ Categórico (2s, 0.5)*
R(1-2)	5	55,97	36,48 (σ =30,54)	55,87 (σ =0,07)
R(1-3)	4	54,96	54,85 (σ =0,94)	54,85 (σ =0,94)
R(1-4)	6	55,04	47,09 (σ =23,26)	54,76 (σ =0,74)
R(1-8)	4	51,78	51,32 (σ =0,50)	51,32 (σ =0,5)
R(2-3)	3	0,67	0,47 (σ =0,26)	0,47 (σ =0,26)
R(2-4)	2	0,75	0,82 (σ =0,00)	0,82 (σ =0,00)
R(2-8)	2	4,02	4,01 (σ =0,21)	4,01 (σ =0,21)
R(3-4)	5	0,08	0,94 (σ =1,63)	0,21 (σ =0,04)
R(3-8)	5	3,59	4,86 (σ =2,85)	3,58 (σ =0,17)
R(4-8)	5	3,34	3,26 (σ =0,24)	3,26 (σ =0,24)

Assim como observado na 2ª campanha, o uso da distribuição categórica permitiu melhorar a precisão dos valores de todas as relações apresentadas na

Tabela 6. Vale observar esse efeito é particularmente destacado para as relações R(1-2) e R(1-4).

A Tabela 7 sumariza mais resultados obtidos na 3ª campanha, mas desta vez, são consideradas relações de sincronização envolvendo os vídeos v_5, v_6 e v_7, todos com conteúdos descontínuos, devido a cortes de edição.

A inserção de cortes faz com que o vídeo produzido seja tratado “fisicamente” como um único conteúdo, mas “logicamente” como um conjunto de segmentos. Como resultado, a tarefa de se encontrar um único ponto para sincronizar o conteúdo de dois vídeos não é suficiente para garantir a apresentação sincronizada destes vídeos a partir deste ponto. De fato, a tarefa passa a ser encontrar um ponto de sincronização entre cada um dos segmentos lógicos do vídeo, em geral, resultantes de cortes de cenas durante a captura ou edição.

Na Tabela 7 um valor de Δ associado a cada um dos segmentos lógico do conteúdo do vídeo “fisicamente contínuo” é apresentado. Em outras palavras, os *workers* são capazes de encontrar valores para Δ_1 e Δ_2 que capturam os diferentes pontos de sincronização entre os vídeos analisados, considerando a segmentação lógica dos conteúdos destes vídeos.

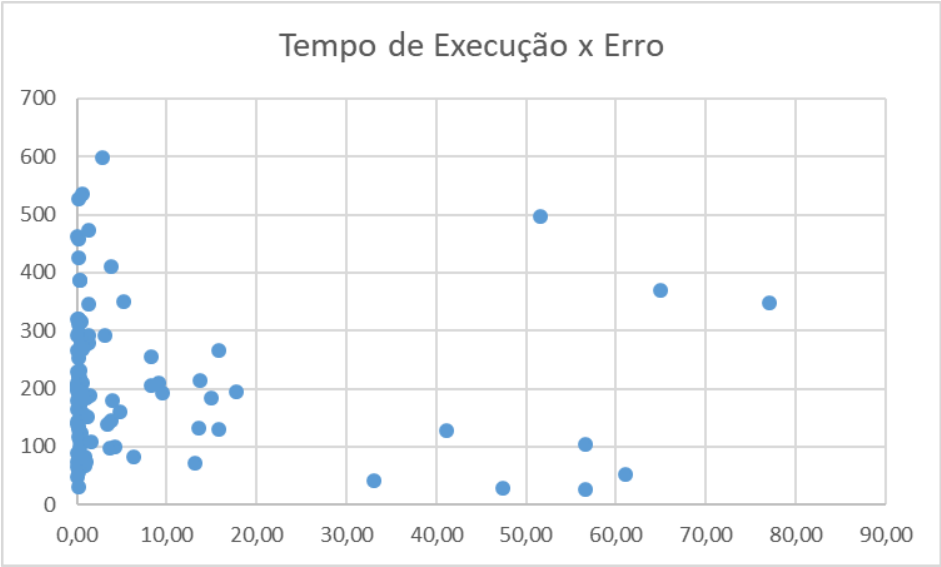
Tabela 7 - Precisão da Campanha 03 (valores absolutos) - B

	N.	*Δ_1*	*Δ_2*	*Δ_1 Categórico (2,0.5)*	*Δ_2 Categórico (2,0.5)*
(1-5)	5	75,3	65,7	74,84	66,69 (σ=0,12)
(1-6)	4	71,57	61,94	**X**	62,57 (σ=0,88)
(1-7)	5	51,55	8,8	**X**	9,26(σ=0,58)
(2-5)	6	19,3	9,8	19,00 (σ=1,07)	5,97 (σ=1,24)
(2-6)	4	15,83	6,1	15,72	0,31 (σ=0,53) **ERRO**
(2-7)	3	46,75	4,00	46,48	4,11 (σ=0,06)
(3-5)	2	20,56	10,16	**X**	9,64 (σ=0,50)
(3-6)	3	16,36	6,76	**16,24**	**6,96** (σ=0,33)
(3-7)	3	46,12	3,37	46,94 (σ=0,35)	**X**
(4-5)	3	20,03	10,43	20,15	8,17
(4-6)	1	16,53	6,90	**X**	8,55
(4-7)	7	46,28	3,53	45,88 (σ=0,24)	3,11
(5-6)	6	3,66	3,66	3,78 (σ=0,57)	3,78 (σ=0,57)
(5-7)	7	56,64	13,89	56,34 (σ=0,30)	0,02 (σ=0,42) **ERRO**
(5-8)	4	23.66	14,04	**X**	14,34 (σ=0,26)
(6-7)	4	53,00	43,25	50,19 (σ=0,57) **ERRO**	39,29 (σ=0,65) **ERRO**
(6-8)	5	20,14	10,54	19,69 (σ=0,90)	**X**
(7-8)	3	42,80	0	41,14	0,15 (σ=0,02)

Entretanto, o algoritmo para convergência apresentado e utilizado inicialmente no experimento, se baseia em pegar os valores da categoria com maior probabilidade, e definir aquele intervalo como o correto. Neste caso, com a descontinuidade dos vídeos ocorre que mais de um intervalo possuirá valores corretos. Sendo assim, a sincronização com vídeos descontínuos pode ter dois valores de convergência. Isto faz com que seja necessário analisar os dois possíveis pontos de sincronização para cada relação, o que é mostrado na Tabela 7 através da presença das colunas Δ_1 e Δ_2 e das convergências categóricas para dois intervalos com mais contribuições. Quando um **X** é apresentado como valor categórico, implica que não ocorreu um segundo intervalo, por não ter tido contribuições em mais de um intervalo, ou por que os vários intervalos possuíram a mesma quantidade de contribuições. Não havendo um segundo intervalo.

Apesar da dificuldade devido aos vídeos descontínuos, parte das contribuições conseguiu identificar corretamente o ponto de sincronização de pelo menos um dos segmentos internos dos vídeos. Porém, a partir dos valores encontrados ficou claro que a precisão na 3ª campanha apresentou uma piora se comparada às duas campanhas anteriores (Figura 47).

Figura 47 - Dispersão das Contribuições da Campanha 03



Fonte: Elaborada pelo autor

Não apenas isso, o número de erros foi maior, principalmente quando analisados os vídeos com descontinuidade com 4 erros. Porém, ainda assim, foi possível encontrar o valor de sincronização para todas as relações, e em alguns casos, os dois valores foram encontrados.

7.3.4. Análise dos Resultados do Experimento Final

As três campanhas mostraram que o uso exclusivo de uma *crowd* real é capaz de gerar a especificação de diferentes perfis de *datasets*, incluindo *UGVs*. Apesar das funcionalidades limitadas para manipulação dos vídeos, a *crowd* conseguiu um nível de sincronização melhor que o esperado e com uma alta taxa de relações encontradas corretamente, inclusive funcionando no primeiro experimento a identificação de vídeos sem sobreposição.

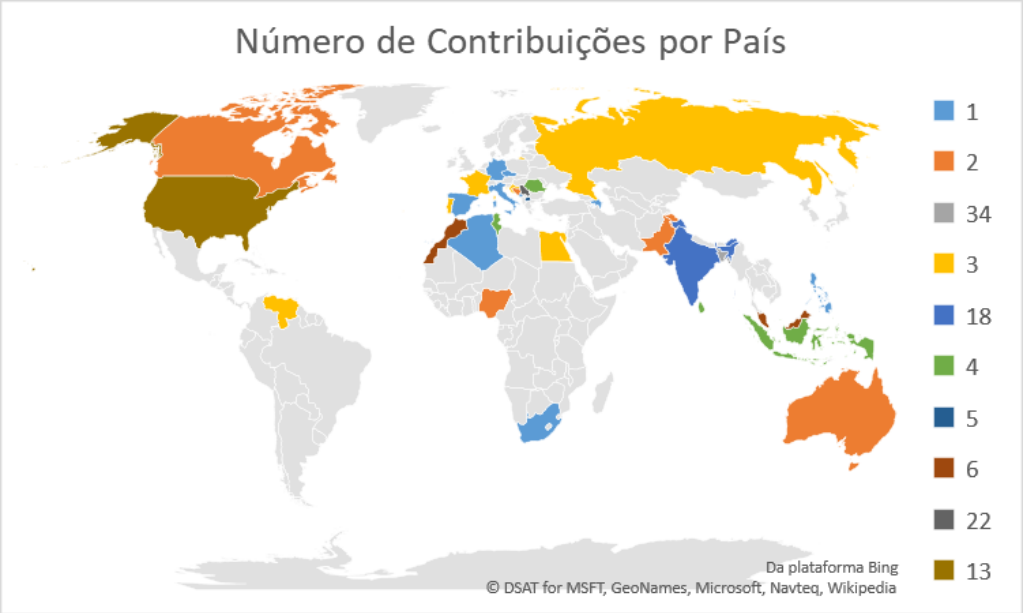
Em todas as campanhas, apenas uma relação convergiu de forma errada, ou seja, para um valor longe do esperado. Nesse caso ficou evidente uma situação inesperada: diante da dificuldade de encontrar um ponto de sincronização, o valor de contribuição (Δ) foi em muitos casos próximo a 0s. Em um cenário difícil, como foi a relação R(2-6) da campanha 03, muitas contribuições foram zeradas, o que levou o algoritmo de convergência a identificar a relação como tendo valor 0. Situação semelhante ocorreu na relação R(2-3) da segunda campanha, onde contribuições nulas impediram a convergência dos valores corretos. Sendo assim, o tratamento da contribuição nula deverá ser resolvido para evitar este tipo de situação.

Um dos diferencias deste experimento em relação aos demais apresentados na execução da pesquisa, está no perfil da *crowd*. A *crowd* utilizada foi contratada a partir do pagamento de pequenos valores para realização das tarefas. Pessoas de todo mundo puderam contribuir com a execução do experimento, como mostrado na Figura 48, com destaque para três países: Bangladesh, Sérvia, Índia e EUA, com respectivamente: 34, 22, 18 e 13 participantes cada.

Por fim, definindo como um resultado de qualidade (confiável) contribuições com um erro menor ou igual a 0,5s, e uma contribuição de baixa qualidade valores maiores que isto, é possível afirmar a partir da análise do erro de todas contribuições nas três campanhas realizadas que a *crowd* utilizada neste experimento teve um nível de confiança de 51,93%.

Outro ponto a ser destacado foi o uso de um *dataset* contendo vídeos correlacionados a um mesmo evento social, mas com descontinuidade lógica em seus conteúdos. Esta descontinuidade tem influência direta no processo de sincronização realizado:

Figura 48 - Mapa Mundo das Contribuições



Fonte: Elaborada pelo autor

- A agregação dos valores das contribuições é prejudicada, pois existe a possibilidade de mais de um Δ para cada relação. Desta forma, podem surgir mais de um agrupamento correto das contribuições feitas, e com isso, uma relação que deveria convergir, poderá divergir devido ao crescimento de dois grupos sendo necessária uma análise diferenciada da convergência para estes casos.
- O grau de esforço do *worker* aumenta com a quebra da continuidade do conteúdo do vídeo, além disso, o *worker* pode ter dificuldade em concluir sua tarefa até perceber que o vídeo está logicamente segmentado, pois provavelmente ficará tentando sincronizar todo o conteúdo do par de vídeos em vão. Por outro lado, alguns worker podem nem perceber a descontinuidade, submetendo o primeiro ponto de sincronização que encontrarem;
- Não há previsão no modelo de sincronização da tese para que vários pontos de sincronização entre vídeos sejam armazenados. No escopo da tese, estes casos deveriam ser tratados como um caso no qual o conjunto de vídeos de entrada é igual ao número de segmentos lógicos a serem processados. Uma alternativa para solucionar este problema, seria pré-processar o vídeo em

busca de cortes, e quando encontrado, transformar cada segmento em um vídeo isolado.

Na apresentação combinada deste tipo de conteúdo, deve haver previsão para que o valor de Δ seja atribuído para cada par de segmentos lógicos apresentados de forma sincronizada.

7.4. CONSIDERAÇÕES

O objetivo deste capítulo foi demonstrar que a *crowd* pode solucionar o problema de sincronizar um conjunto de vídeos correlacionados. Neste caso, cada *worker* executa uma tarefa que consiste em processar um par de vídeos para encontrar um ponto de sincronização entre seus conteúdos.

Três experimentos foram propostos para avaliar a proposta de sincronização pela *crowd*. Os experimentos evolveram ora uma *crowd* controlada voluntária, recrutada de maneira informal, a partir de um grupo de pessoas próximas ao autor da tese, ora uma *crowd* recrutada, gerenciada e recompensada com apoio da plataforma *online* de *crowdsourcing MicroWorkers*. Neste caso, a ideia foi verificar se a *crowd* contratada alcançaria um grau de precisão maior que uma *crowd* controlada e voluntária.

O primeiro experimento envolveu uma *crowd* controlada voluntária realizando testes em laboratório para verificar a viabilidade de uma abordagem baseada na observação de segmentos de 5s, extraídos dos vídeos a serem sincronizados. Esta abordagem apresentou problemas com relação ao número excessivo de tarefas necessárias para realizar a sincronização e a grande possibilidade de insucesso na execução da tarefa, que pode desestimular a participação dos *workers*.

O experimento seguinte buscou mesclar o processamento da *crowd* com um processamento automático para alcançar a sincronização dos vídeos correlacionados. A conclusão obtida foi que o uso de uma técnica de processamento automático do conteúdo dos vídeos reduz, efetivamente, o esforço realizado pela *crowd* na sincronização. Dependendo das características dos vídeos, o método automático é capaz de sincronizar todos os vídeos, mas, à medida que os conteúdos dos vídeos apresentam maior dificuldade para a sincronização, a *crowd* consegue contribuir no processo, identificando falhas e sincronizando vídeos. Apesar dos

workers terem conseguido concluir as tarefas propostas, um número relativamente alto de contribuições foi necessário, principalmente na fase de validação, onde todas as relações obtidas automaticamente tiveram que ser analisadas uma a uma. Assim como é feito na etapa de sincronização, com o auxílio dos métodos de inferência, a validação não deve ter que verificar todas relações, apenas algumas que possam servir de base para calcular as demais relações, e assim, automaticamente validar as demais. A abordagem híbrida foi elaborada para tentar tornar os métodos complementares: a *crowd* garante a verificação da especificação e cuida das exceções que os métodos automáticos não podem lidar. Por outro lado, os métodos automáticos evitam um *cold start* para a sincronização do *crowdsourcing*.

O “Experimento Final” utilizou uma *crowd* com perfil desconhecido, espalhada pelo mundo e recompensada através de ganho financeiro para sincronizar diversos vídeos. Esse experimento mostrou que mesmo uma *crowd* com perfil desconhecido é capaz de gerar a especificação de sincronização para os conjuntos de vídeos utilizados. Alguns problemas das ferramentas utilizadas ficaram claros, como a atribuição do valor 0 por vários usuários quando estes não encontraram um ponto de sincronização. Algumas limitações esperadas ficaram comprovadas, como mostrado na terceira campanha, na qual o uso de vídeos com cortes no conteúdo dificultou o trabalho da crowd e a convergência dos valores de sincronização gerados pelos *workers*.

8. CONSIDERAÇÕES FINAIS

A sincronização multimídia permite a criação de apresentações enriquecidas, nas quais diferentes entidades são combinadas dentro de uma apresentação com o objetivo de melhorar a experiência de quem a assiste. Assim, sincronização é o ato de fazer com que diferentes entidades funcionem de forma combinada, numa mesma cadência, como se fossem uma entidade única.

A sincronização pode ser aplicada a um grupo de vídeos correlacionados para a criação de mosaicos de vídeos, permitindo, por exemplo, chavear a qualquer instante o ângulo de visualização de um evento ou ainda, para recriar a história de um evento social a partir de um conjunto de vídeos gerados de forma voluntária, independente e descoordenada por usuários. Estes vídeos, denominados na tese de *User Generated Videos* ou *UGVs*, possuem características particulares que dificultam sua sincronização, tais como: múltiplos usuários podem, usando suas câmeras pessoais, registrar diferentes pontos de vista de um mesmo evento, com diferentes configurações de câmera, duração e outras características persistentes aos vídeos. Nesse modelo de captura, os usuários não estão sendo coordenados, assim não se sabe qual parte do evento o usuário capturou em sua câmera. Através da sincronização, é possível transmitir vários fluxos de vídeo capturados em tempo real a partir de diferentes ângulos e perspectivas ou usar os vídeos resultantes desse processo para recontar, posteriormente, a história do evento, desde que seja possível encontrar pontos de sincronização entre os conteúdos destes vídeos.

Diferentes abordagens podem ser utilizadas para sincronizar *UGVs*. A fim de superar as limitações dos métodos automáticos para processar esses conteúdos, esta tese propôs e avaliou o uso de um método para solucionar o problema de sincronizar um conjunto de vídeos correlacionados a um mesmo evento social usando o modelo *crowdsourcing*.

A tese mostrou como usuários finais podem ser utilizados para solucionar o problema de sincronizar localmente, no destino de apresentação, fluxos de vídeo correlacionados, distribuídos, de diferentes fontes não sincronizadas. O problema foi resolvido com um método que aplica a computação humana para definir o alinhamento temporal dos fluxos de vídeo com o uso de acopladores temporais, que armazenam o valor do atraso verificado entre os fluxos correlacionados dos vídeos.

O desafio da sincronização de *UGVs* não está apenas na questão de alinhar no tempo, os diferentes fluxos vídeos provenientes de múltiplas fontes. Os conteúdos dos *UGVs* são, por definição, heterogêneos, dificultando (ou mesmo, impossibilitando) a aplicação de técnicas automáticas de sincronização. O método baseado em *crowdsourcing* tenta preencher exatamente essa lacuna.

Conforme foi avaliado nos experimentos relatados no texto, o uso da *crowd* permite contornar algumas das limitações das técnicas atuais de sincronização baseadas na extração de informações “de baixo nível”, tais como cores de pixels, intensidade de som, vetores de movimento, etc. Estas técnicas não conseguem um bom desempenho nas situações de captura com câmeras móveis, fundos em constante mudança, objetos sobrepostos entre outras situações.

Para solucionar o problema de sincronizar *UGVs*, é necessário fazer com que a *crowd* encontre os pontos de sincronização entre os vídeos de um conjunto de entrada. A solução do problema é alcançada com a agregação dos valores de contribuições obtidas com a execução de tarefas “simples” pelos *workers*. Duas abordagens para a definição destas tarefas foram apresentadas, sendo a primeira baseada em segmentos extraídos dos vídeos a serem processados e a segunda, baseada no uso de *keyframes* representando o conteúdo de trechos destes vídeos. As avaliações experimentais revelaram falhas em ambas as abordagens, como a necessidade de muitas contribuições, perda da continuidade e da percepção do movimento nas cenas para facilitar a inspeção do conteúdo dos vídeos. Com isto, diversos experimentos em seguida permitiram ao *worker* analisar o vídeo completo para encontrar o ponto de sincronização, mas dando a ele a possibilidade de navegar livremente nos vídeos sem precisar assisti-lo por completo.

Também foi observado que o processo de sincronização baseado em *crowdsourcing* introduz novas questões a serem respondidas, dentre as quais: como convocar a *crowd* a contribuir; como validar tarefas realizadas por eles; e como integrar as contribuições. Estes não são problemas exclusivos de usar *crowdsourcing* junto ao processamento de vídeos, mas o uso de *crowdsourcing* em geral.

A *crowd* pode ser recrutada para executar tarefas, as quais, após concluídas, geram uma recompensa aos *workers*. O tipo de recompensa pode variar entre pagamentos, a competição (gamificação) ou a simples disposição em ajudar. Na

pesquisa, tanto foram utilizadas *crowds* pagas e voluntárias para que fosse possível analisar os resultados obtidos *versus* recompensas oferecidas, além de mostrar que o sistema de *crowdsourcing* implementado no contexto desta tese poderia ser integrado a uma plataforma comercial de gestão de *workers*.

Os experimentos realizados *utilizando* a *crowd* em diferentes situações foram apresentados no penúltimo capítulo desta tese. Inicialmente, o resultado de um experimento realizado em laboratório foi apresentado, para avaliar uma abordagem alternativa para as tarefas realizadas pelos *workers*. Em seguida, um experimento com o uso de um método híbrido foi conduzido. O experimento demonstrou que o sistema híbrido implementado, no qual a *crowd* trabalha sobre os resultados obtidos pelos métodos automáticos, apresenta desempenho superior às abordagens isoladas. Por fim, uma *crowd* é recrutada, a partir de uma chamada com oferta de pagamento, a sincronizar diferentes conjuntos de vídeos para evidenciar o uso do sistema *crowdsourcing* implementado.

8.1.REVISITANDO AS QUESTÕES DE PESQUISA

No primeiro capítulo, quatro questões de pesquisa que nortearam o desenvolvimento desta tese foram apresentadas. Ao longo dos capítulos, o leitor pôde encontrar algumas das respostas de tais perguntas. Nesta seção, estas perguntas serão revisitadas e uma sumarização das respostas associadas a cada uma delas durante o texto será realizada.

1. *Como sincronizar UGVs, considerando um cenário distribuído e falta de informações de sincronização explícita destes vídeos?*

No capítulo 2 foi apresentado o processo de alinhamento de *timelines*, no qual vídeos provenientes de diversas fontes podem ser sincronizados no dispositivo do usuário final, mesmo que as fontes destes vídeos sejam assíncronas e que o tempo de transmissão dos vídeos até o usuário seja desconhecido. Isto é possível com o uso dos acopladores que armazenam informações de sincronização de forma independente das fontes e dos atrasos de transmissão. Então, utilizando os acopladores é possível realizar a sincronização e apresentação síncrona dos vídeos gerados por usuários.

2. *É possível sincronizar UGVs com o uso de técnicas de crowdsourcing?*

No capítulo 3, todas as informações necessárias para a compreensão do conceito de *crowdsourcing* foram apresentadas. O entendimento de como os vídeos podem ser processados pela *crowd* formaram a base para a adaptação do *crowdsourcing* à sincronização de *UGVs* proposta no capítulo 4. Em resumo, os capítulos 3 e 4 apresentam em sua essência a teoria por trás do uso de *crowdsourcing* para a sincronização, sendo esta teoria complementada pela prática através dos experimentos do capítulo 7, mostrando que é possível que a *crowd* efetivamente sincronize *UGVs*.

3. *Como é definido este método de sincronização de vídeos utilizando a crowd?*

No capítulo 5 é apresentado ao leitor o *CrowdVideo*. Apesar de ser generalizado para processamento vídeos pela *crowd*, ele foi especificado a partir dos conhecimentos adquiridos na pesquisa teórica quanto no desenvolvimento prático das aplicações de sincronização. O modelo mostra exatamente a sincronização de *UGVs* pela *crowd* pode ser realizada. Porém, apenas a apresentação do modelo não foi suficiente para deixar claro todas as características do método. Assim, como mostrado no capítulo 6, foi realizada a simulação do método de sincronização utilizando como base o *CrowdVideo* instanciado pelo problema de sincronização. Nesta simulação são mostrados os impactos que diferentes variáveis como confiabilidade da *crowd* e intervalo de categorização podem ter na sincronização, por exemplo, gerando a necessidade de mais contribuições para solução do problema.

4. *O uso de uma abordagem híbrida pode auxiliar o problema da sincronização dos vídeos pela crowd?*

No capítulo 7, em conjunto com outros experimentos, foi apresentada a solução híbrida para a sincronização de vídeos, onde tanto a computação humana quanto a tradicional são utilizadas juntas para a solução do problema de sincronização. O uso dessa solução mostrou que a *crowd* é capaz de melhorar os resultados da sincronização obtidos por um método automático, aumentando o número de relações encontradas e refinando relações que não tiveram uma sincronia fina alcança pelo método automático. Apesar destas melhoras, o método aplicado deve ser melhorado, pois muitas contribuições em algumas etapas foram necessárias para o processamento pela *crowd*, e especificamente na etapa de identificação de falhas de sincronização os testes mostraram que muitos defeitos passaram pelo filtro

da *crowd*, resultando em uma pequena melhora, diante do que poderia ter sido alcançado.

8.2.PUBLICAÇÕES RELACIONADAS À TESE

No decorrer da pesquisa, foram realizadas publicações como parte dos resultados obtidos. A seguir são listadas as publicações alcançadas até o momento.

O início da pesquisa focou no estudo de técnicas de sincronização de múltiplos conteúdos. Neste contexto, os seguintes trabalhos foram publicados, em ordem cronológica:

- **SEGUNDO, RICARDO MENDES COSTA**; SANTOS, CELSO ALBERTO SAIBEL. Second screen event flow synchronization. In: 2013 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), 2013, London, UK.
- **COSTA SEGUNDO, RICARDO MENDES**; SANTOS, CELSO ALBERTO SAIBEL. Systematic Review of Multiple Contents Synchronization in Interactive Television Scenario. ISRN Communications and Networking, v. 2014, p. 1-17, 2014.
- **SEGUNDO, RICARDO MENDES COSTA**; SANTOS, CELSO ALBERTO SAIBEL. Timeline Alignment of Multiple TV Contents. In Proceedings of the 20th Brazilian Symposium on Multimedia and the Web - WebMedia '14, 2014. p. 195.
- **SEGUNDO, RICARDO MENDES COSTA**; SANTOS, CELSO ALBERTO SAIBEL. Remote Temporal Couplers for Multiple Content Synchronization. In: 2015 IEEE International Conference on Computer and Information Technology. 2015 p. 532.

Em seguida o foco se virou ao uso da *crowd* no processo de sincronização. Os artigos publicados sobre este tema fora:

- **SEGUNDO, RICARDO MENDES COSTA**; de AMORIM, MARCELO NOVAES; SANTOS, CELSO ALBERTO SAIBEL. LiveSync: a Tool for Real Time Video Streaming Synchronization from Independent Sources. In: WebMedia 2016, 2016, Teresina. Workshop de Ferramentas e Aplicações, 2016.
- **SEGUNDO, RICARDO MENDES COSTA**, de AMORIM, MARCELO NOVAES, SANTOS, CELSO ALBERTO SAIBEL. "LiveSync: A Method for Real Time Video

Streaming Synchronization from Independent Sources." *Journal of Information and Data Management, 2017.*

- **SEGUNDO, RICARDO MENDES COSTA**; de AMORIM, MARCELO NOVAES; SANTOS, CELSO ALBERTO SAIBEL. Crowdsourcing & Multimedia. In: the 22nd Brazilian Symposium, 2016, Teresina. Proceedings of the 22nd Brazilian Symposium on Multimedia and the Web - Webmedia '16, 2016. p. 11.
- **SEGUNDO, RICARDO MENDES COSTA**, de AMORIM, MARCELO NOVAES, and SANTOS, CELSO ALBERTO SAIBEL. "CrowdSync: User generated videos synchronization using crowdsourcing." *Systems, Signals and Image Processing (IWSSIP), 2017 International Conference on*. IEEE, 2017.

Alem de trabalhos resultantes diretamente da pesquisa sobre *crowdsourcing* e sincronização, ocorreram em trabalhos de outros membros do laboratório:

- LEMOS, VICTOR S.; FERREIRA, RAFAEL F.; **SEGUNDO, RICARDO MENDES COSTA** ; COSTALONGA, LEANDRO L.; SANTOS, SANTOS, CELSO ALBERTO SAIBEL. Local Synchronization of Web Applications with Audio Markings. Proceedings of the 22nd Brazilian Symposium on Multimedia and the Web - Webmedia '16, 2016. p. 159.
- de AMORIM, MARCELO NOVAES; SANTOS, CELSO ALBERTO SAIBEL; **SEGUNDO, RICARDO MENDES COSTA**; ORIVALDO TAVARES. Video Annotation by Cascading Microtasks: a Crowdsourcing Approach. 23rd Brazilian Symposium on Multimedia and the Web - Webmedia '17

8.3.TRABALHOS FUTUROS

Apesar dos vários resultados obtidos, melhorias e novas contribuições podem ser obtidas a partir do estado atual do trabalho:

O primeiro ponto a ser explorado está relacionado às tarefas realizadas pela *crowd*. Apesar de apresentar algumas abordagens desenvolvidas, como o uso de frames, segmentos, e dos vídeos em sua completa extensão, mas com auxílio A sua navegação, não foi realizada uma análise detalha de cada método em separado, mas apenas seu uso como ferramenta para atingir a sincronização dos vídeos. É necessário executar as diferentes técnicas e avaliar sua usabilidade, além de

verificar a precisão que cada técnica pode alcançar, levando-se em conta também o custo de cada tarefa;

Os diferentes formatos de tarefas utilizados podem ter impacto em um fator que não foi considerado na pesquisa: o uso da rede. Ao trabalhar com vídeos, o *worker* deve receber vídeos, segmentos ou frames para que seja possível a sincronização. Será mais válido trabalhar com vídeos completos hospedados em *sites* que cuidam de todo processo de distribuição dos vídeos, com vídeos adaptativos e tudo mais? Ou seria melhor trabalhar com trechos dos vídeos usando uma plataforma própria de processamento? A qualidade do vídeo transmitido tem impacto na realização das tarefas?

O método híbrido foi usado em um experimento que integrou técnicas automáticas e *crowdsourcing* na sincronização dos vídeos. Porém, ainda é preciso integrar todas as etapas do método em uma plataforma única. Devido ao uso de uma ferramenta paga na etapa automática, não foi possível sua integração completa à plataforma, sendo necessária a geração de um arquivo temporário para tonar possível a comunicação. O método híbrido também teve limitações na fase de validação, exigindo a verificação de todas as relações, uma a uma, algo que deve ser melhorado. Além disto, a *crowd* deixou de identificar relações que possuíam problemas, indicando que a fase de validações das sincronizações criadas deve ser melhorada;

O foco da pesquisa foi a sincronização com o uso de *crowdsourcing* para encontrar pontos de sincronização entre os conteúdos dos vídeos. Porém, a sincronização dos pontos depende de outras etapas dentro do processo (por exemplo, o agrupamento dos vídeos correlacionados), as quais realizadas de forma errada podem comprometer o resultado final do processo. Nos trabalhos futuros, é preciso integrar a sincronização apresentada com outras etapas do processo que não foram abordadas, criando processo mais preciso de sincronização de *UGVs*.

Outro ponto em aberto no processo é o uso da especificação de sincronização criada. Apesar de citar o uso da seleção de vídeo e áudio personalizados para criar uma apresentação, a apresentação em mosaico dos vídeos, ou uso de cortes entre os vídeos para construção de uma apresentação, mais estudos devem ser realizados sobre as formas de apresentação de múltiplos *UGVs* sincronizados.

Referências

AGRESTI, A. A. M. K. Categorical data analysis. International encyclopedia of statistical science. [S.l.]: Springer Berlin Heidelberg, 2011.

AHN, L. V. Three human computation projects. Proceedings of the 42nd ACM technical symposium on Computer science education, 2011.

ALBARQOUNI, S. E. A. Aggnet: deep learning from crowds for mitosis detection in breast cancer histology images. IEEE transactions on medical imaging, 2016. 1313-1321.

ALGUR, S. P.; BHAT, P. Web Video Object Mining: Expectation Maximization and Density Based Clustering of Web Video Metadata Objects. International Journal of Information Engineering and Electronic Business, v. 8, p. 69, 2016.

ALILOUPOUR, N. P. The Impact of Technology on the Entertainment Distribution Market: The Effects of Netflix and Hulu on Cable Revenue. Claremont Colleges. [S.l.]. 2016.

AMORIM, M. et al. Video Annotation by Cascading Microtasks: a Crowdsourcing Approach. Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia). Gramado: [s.n.]. 2017.

ANEGEKUH, L.; SUN, L.; IFEACHOR, E. A screening methodology for crowdsourcing video QoE evaluation. Global Communications Conference (GLOBECOM), 2014 IEEE. [S.l.]: [s.n.]. 2014. p. 1152-1157.

ARAN, O.; BIEL, J.-I.; GATICA-PEREZ, D. Broadcasting oneself: Visual discovery of vlogging styles. Multimedia, IEEE Transactions on, v. 16, p. 201-215, 2014.

ARAÚJO, D. A. T. M. U. et al. An approach to generate and embed sign language video tracks into multimedia contents. Information Sciences, v. 281, p. 762-780, 2014.

AREV, I. E. A. Automatic editing of footage from multiple social cameras. ACM Transactions on Graphics (TOG), 2014.

BABA, A. et al. Seamless, synchronous, and supportive: welcome to hybridcast: an advanced hybrid broadcast and broadband system. Consumer Electronics Magazine, IEEE, v. 1, p. 43-52, 2012.

BANO, S.; CAVALLARO, A. Discovery and organization of multi-camera user-generated videos of the same event. Information Sciences, v. 302, p. 108-121, 2015.

BAVEYE, Y. et al. LIRIS-ACCEDE: A video database for affective content analysis. Affective Computing, IEEE Transactions on, v. 6, p. 43-55, 2015.

BERTINI, M. et al. Socially-aware video recommendation using users´profiles and crowdsourced annotations. Proceedings of the 2nd international workshop on Socially-aware multimedia. [S.l.]: [s.n.]. 2013. p. 13-18.

BHIMANI, J. et al. Vox populi: enabling community-based narratives through collaboration and content creation. Proceedings of the 11th european conference on Interactive TV and video. [S.l.]: [s.n.]. 2013. p. 31-40.

BIEL, J.-I.; GATICA-PEREZ, D. The youtube lens: Crowdsourced personality impressions and audiovisual analysis of vlogs. Multimedia, IEEE Transactions on, v. 15, p. 41-55, 2013.

BLAKOWSKI, G.; STEINMETZ, R. A media synchronization survey: reference model, specification, and case studies. IEEE journal on selected areas in communications, v. 14, p. 5-35, 1996.

BOHEZ, S. et al. Management of crowdsourced first-person video: street view live. Proceedings of the 13th International Conference on Mobile and Ubiquitous Multimedia. [S.l.]: [s.n.]. 2014. p. 11-19.

BRABHAM, D. C. et al. Crowdsourcing applications for public health. American journal of Preventive Medicine, v. 46, p. 179-187, 2014.

BURMANIA, A.; PARTHASARATHY, S.; BUSSO, C. Increasing the Reliability of Crowdsourcing Evaluations Using Online Quality Assessment. IEEE Transactions on Affective Computing, v. 7, p. 374-388, Oct 2016. ISSN ISSN: 1949-3045.

BYWOOD, L. . P. G. T. E. Embracing the threat: machine translation as a solution for subtitling, v. 25, p. 492-508, 2017.

CARLIER, A. et al. 3D Interest Maps From Simultaneous Video Recordings. Proceedings of the ACM International Conference on Multimedia. [S.l.]: [s.n.]. 2014. p. 577-586.

CAVALCANTI, M.; NEPOMUCENO, C. O conhecimento em rede: como implantar projetos de inteligência coletiva. Elsevier, 2017.

CESAR, P.; CHORIANOPOULOS, K. The evolution of TV systems, content, and users toward interactivity. Foundations and Trends in Human-Computer Interaction, v. 2, p. 373-95, 2009.

CESAR, P.; GEERTS, D. Past, present, and future of social TV: A categorization. Consumer Communications and Networking Conference (CCNC), 2011 IEEE. [S.l.]: [s.n.]. 2011. p. 347-351.

CHEN, F. et al. Cloud-Assisted Live Streaming for Crowdsourced Multimedia Content. Multimedia, IEEE Transactions on, v. 17, p. 1471-1483, 2015.

CHEN, S. et al. Crowd map: Accurate reconstruction of indoor floor plans from crowdsourced sensor-rich videos. Distributed Computing Systems (ICDCS), 2015 IEEE 35th International Conference on. [S.l.]: [s.n.]. 2015. p. 1-10.

CHORIANOPOULOS, K. Crowdsourcing user interactions with the video player. Proceedings of the 18th Brazilian symposium on Multimedia and the web. [S.l.]: [s.n.]. 2012. p. 13-16.

CISCO. Forecast and Methodology, 2016–2021. [S.l.]: [s.n.]. 2017.

CRAGGS, B.; KILGALLON SCOTT, M.; ALEXANDER, J. ThumbReels: query sensitive web video previews based on temporal, crowdsourced, semantic tagging. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. [S.l.]: [s.n.]. 2014. p. 1217-1220.

DESELL, T. et al. On the effectiveness of crowd sourcing avian nesting video analysis at Wildlife@ Home. Procedia Computer Science, v. 51, p. 384-393, 2015.

DESHPANDE, R. et al. A crowdsourcing caption editor for educational videos. Frontiers in Education Conference (FIE), 2014 IEEE. [S.l.]: [s.n.]. 2014. p. 1-8.

DI SALVO, R.; GIORDANO, D.; KAVASIDIS, I. A crowdsourcing approach to support video annotation. Proceedings of the International Workshop on Video and Image Ground Truth in Computer Vision Applications. [S.l.]: [s.n.]. 2013. p. 8.

DIFALLAH, D. E. E. A. The dynamics of micro-task crowdsourcing: The case of amazon mturk. Proceedings of the 24th International Conference on World Wide Web, 2015.

DOAN, A.; RAMAKRISHNAN, R.; HALEVY, A. Y. Crowdsourcing systems on the world-wide web. Communications of the ACM, v. 54, p. 86-96, 2011.

DOMMEL, H.-P.; VERMA, S. K. Multipoint synchronization protocol. Systems, Man and Cybernetics, 2004 IEEE International Conference on. [S.l.]: [s.n.]. 2004. p. 4631-4635.

DOUZE, M. et al. Circulant temporal encoding for video retrieval and temporal alignment.

DYBA, T.; DINGSOYR, T.; HANSSEN, G. K. Applying systematic reviews to diverse study types: An experience report. null. [S.l.]: [s.n.]. 2007. p. 225-234.

EGGER, S. et al. The impact of adaptation strategies on perceived quality of http adaptive streaming. Proceedings of the 2014 Workshop on Design, Quality and Deployment of Adaptive Video Streaming. [S.l.]: [s.n.]. 2014. p. 31-36.

ENCELLE, B. et al. Annotation-based video enrichment for blind people: A pilot study on the use of earcons and speech synthesis. The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility. [S.l.]: [s.n.]. 2011. p. 123-130.

ESTELLÉS, E.; GONZÁLEZ, F. Towards an integrated crowdsourcing definition. Journal of Information science, v. 38, p. 189-200, 2012.

FAN, J. E. A. A hybrid machine-crowdsourcing system for matching web tables. IEEE 30th International Conference on Data Engineering (ICDE), 2014.

FERRACANI, A. et al. A System for Video Recommendation using Visual Saliency, Crowdsourced and Automatic Annotations. Proceedings of the 23rd Annual ACM Conference on Multimedia Conference. [S.l.]: [s.n.]. 2015. p. 757-758.

FINK, M.; COVELL, M.; BALUJA, S. Social-and interactive-television applications based on real-time ambient-audio identification. Proceedings of EuroITV. [S.l.]: [s.n.]. 2006. p. 138-146.

FREIBURG, B.; KAMPS, J.; SNOEK, C. G. M. Crowdsourcing visual detectors for video search. Proceedings of the 19th ACM international conference on Multimedia. [S.l.]: [s.n.]. 2011. p. 913-916.

FREMUTH, A.; ADZIC, V.; KALVA, H. Parameterized framework for the analysis of visual quality assessments using crowdsourcing. IS\&T/SPIE Electronic Imaging. [S.l.]: [s.n.]. 2015. p. 93940C--93940C.

FREY, N.; ANTONE, M. Grouping Crowd-Sourced Mobile Videos for Cross-Camera Tracking. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. [S.l.]: [s.n.]. 2013. p. 800-807.

GADGIL, N. J. et al. A web-based video annotation system for crowdsourcing surveillance videos. IS\&T/SPIE Electronic Imaging. [S.l.]: [s.n.]. 2014. p. 90270A--90270A.

GALTON, F. Vox populi (The wisdom of crowds). Nature, 1907.

GARDLO, B. et al. Crowdsourcing 2.0: Enhancing execution speed and reliability of web-based QoE testing. Communications (ICC), 2014 IEEE International Conference on. [S.l.]: [s.n.]. 2014. p. 1070-1075.

GILBERT, A.; BOWDEN, R. iGroup: Weakly supervised image and video grouping. Computer Vision (ICCV), 2011 IEEE International Conference on. [S.l.]: [s.n.]. 2011. p. 2166-2173.

GÖRANSSON, R.; AYDEMIR, A.; JENSFELT, P. Kinect@ Home: Crowdsourced RGB-D data. Proc. of the 2002 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS'02): Cloud Robotics Workshop. [S.l.]: [s.n.]. 2013.

GOTTLIEB, L. et al. Pushing the limits of mechanical turk: qualifying the crowd for video geo-location. Proceedings of the ACM multimedia 2012 workshop on Crowdsourcing for multimedia. [S.l.]: [s.n.]. 2012. p. 23-28.

GUGGENBERGER, M. Aurio: Audio Processing, Analysis and Retrieval. Proceedings of the 23rd ACM international conference on Multimedia. [S.l.]: [s.n.]. 2015.

GUIMARÃES, R. L. et al. Creating personalized memories from social events: community-based support for multi-camera recordings of school concerts. Proceedings of the 19th ACM international conference on Multimedia. Scottsdale, Arizona, USA: [s.n.]. 2011. p. 303-312.

GUO, X. H. G. A. H. W. Image Clustering Based on the Human Intelligence. 10th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), 2015.

HAAS, D. . A. J. . G. L. . &. M. A. Argonaut: macrotask crowdsourcing for complex data processing. Proceedings of the VLDB Endowment, 2015.

HALVEY, M. et al. ViGOR: a grouping oriented interface for search and retrieval in video libraries. Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries. [S.l.]: [s.n.]. 2009. p. 87-96.

HAN, C.-H.; LEE, J.-S. Quality assessment of on-line videos using metadata. Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on. [S.l.]: [s.n.]. 2014. p. 1385-1388.

HAUTZ, J. et al. Let Users Generate Your Video Ads? The Impact of Video Source and Quality on Consumers´Perceptions and Intended Behaviors. Journal of Interactive Marketing, v. 28, p. 1-15, 2014.

HEILBRON, F. C.; NIEBLES, J. C. Collecting and annotating human activities in web videos. Proceedings of International Conference on Multimedia Retrieval. [S.l.]: [s.n.]. 2014. p. 377.

HENTER, D.; BORTH, D.; ULGES, A. Tag suggestion on youtube by personalizing content-based auto-annotation. Proceedings of the ACM multimedia 2012 workshop on Crowdsourcing for multimedia. [S.l.]: [s.n.]. 2012. p. 41-46.

HOSSEINI, M. E. A. The four pillars of crowdsourcing: A reference mode. 2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS), 2014.

HOßFELD, T. et al. Quantification of YouTube QoE via crowdsourcing. Multimedia (ISM), 2011 IEEE International Symposium on. [S.l.]: [s.n.]. 2011. p. 494-499.

HOßFELD, T. et al. Best practices for QoE crowdtesting: QoE assessment with crowdsourcing. Multimedia, IEEE Transactions on, v. 16, p. 541-558, 2014.

HOWE, J. The Rise of Crowdsourcing. Wired magazine, 2006.

HOWSON, C. et al. Second screen TV synchronization. Consumer Electronics-Berlin (ICCE-Berlin), 2011 IEEE International Conference on. [S.l.]: [s.n.]. 2011. p. 361-365.

HUANG, Z.; NAHRSTEDT, K.; STEINMETZ, R. Evolution of temporal multimedia synchronization principles: A historical viewpoint. ACM TOMM, 2013.

HUBERMAN, B. A.; ROMERO, D. M.; WU, F. Crowdsourcing, attention and productivity. Journal of Information Science, 2009.

HUNG, N. Q. V.; AL., E. An evaluation of aggregation techniques in crowdsourcing. International Conference on Web Information Systems Engineering. Berlin: [s.n.]. 2013.

ISHIBASHI, Y.; TSUJI, A.; TASAKA, S. A group synchronization mechanism for stored media in multicast communications. INFOCOM'97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution., Proceedings IEEE. [S.l.]: [s.n.]. 1997. p. 692-700.

JAIN, P. et al. FOCUS: clustering crowdsourced videos by line-of-sight. Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems. [S.l.]: [s.n.]. 2013. p. 8.

JISUP HONG, C. F. B. How good is the crowd at real WSD? Proceedings of the 5th linguistic annotation workshop. Association for Computational Linguistics, 2011.

JOHN P. RULA, V. N. F. E. B. R. B. S. G. No "one-size fits all": towards a principled approach for incentives in mobile crowdsourcing. In Proceedings of the 15th Workshop on Mobile Computing Systems and Applications, 2014.

KACORRI, H.; SHINKAWA, K.; SAITO, S. Introducing game elements in crowdsourced video captioning by non-experts. Proceedings of the 11th Web for All Conference. [S.l.]: [s.n.]. 2014. p. 29.

KATO, K. et al. Generation of a Video Summary on a News Topic Based on SNS Responses to News Stories. Proceedings of the Fourth International Workshop on Crowdsourcing for Multimedia. [S.l.]: [s.n.]. 2015. p. 21-26.

KAZAI, G. . K. J. . &. M.-F. N. Worker types and personality traits in crowdsourcing relevance labels. Proceedings of the 20th ACM international conference on Information and knowledge management. [S.l.]: [s.n.]. 2011.

KEIMEL, C. et al. Video quality evaluation in the cloud. International Packet Video Workshop). [S.l.]: [s.n.]. 2012. p. 155-160.

KEIMEL, C.; HABIGT, J.; DIEPOLD, K. Challenges in crowd-based video quality assessment. Quality of Multimedia Experience (QoMEX), 2012 Fourth International Workshop on. [S.l.]: [s.n.]. 2012. p. 13-18.

KHOSLA, A. et al. Large-scale video summarization using web-image priors. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.]: [s.n.]. 2013. p. 2698-2705.

KIM, G.; SIGAL, L.; XING, E. Joint summarization of large-scale collections of web images and videos for storyline reconstruction. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.]: [s.n.]. 2014. p. 4225-4232.

L. PU, X. C. J. X. X. F. Crowdlet: Optimal worker recruitment for self-organized mobile crowdsourcing. IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications, 2016.

LASECKI, W. S. et al. Answering visual questions with conversational crowd assistants. Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility. [S.l.]: [s.n.]. 2013. p. 18.

LAW, E.; AHN, L. V. Human computation. Synthesis Lectures on Artificial Intelligence and Machine Learning, v. 5, p. 1-121, 2011.

LE, J. et al. Ensuring quality in crowdsourced search relevance evaluation: The effects of training question distribution. SIGIR 2010 workshop on crowdsourcing for search evaluation. [S.l.]: [s.n.]. 2010. p. 21-26.

LEMOS, V. S. . F. R. F. . C. S. R. M. . C. L. L. . &. S. C. A. Local Synchronization of Web Applications with Audio Markings. 22nd Brazilian Symposium on Multimedia and the Web. Teresina: [s.n.]. 2016.

LÉVY, P. Les technologies de l'intelligence: l'avenir de la pensée à l'ère informatique. Editions la découverte, 1993.

MALHOTRA, R. Hybrid broadcast broadband TV: the way forward for connected TVs. Consumer Electronics Magazine, IEEE, v. 2, p. 10-16, 2013.

MARQUES NETO, M. C.; SANTOS, C. A. S. StoryToCode: a new model for specification of convergent interactive digital TV applications. Journal of the Brazilian Computer Society, 2010. 215-227.

MEYER, T.; EFFELSBERG, W.; STEINMETZ, R. A taxonomy on multimedia synchronization. Distributed Computing Systems, 1993., Proceedings of the Fourth Workshop on Future Trends of. [S.l.]: [s.n.]. 1993. p. 97-103.

MO, L. E. A. Optimizing plurality for human intelligence tasks. Proceedings of the 22nd ACM international conference on Information & Knowledge Management, 2013.

MONTAGUD, M.; BORONAT, F. Enhanced adaptive RTCP-based Inter-Destination Multimedia Synchronization approach for distributed applications. Computer Networks, v. 56, p. 2912-2933, 2012.

ORDELMAN, R. J. F. et al. Defining and evaluating video hyperlinking for navigating multimedia archives. Proceedings of the 24th International Conference on World Wide Web Companion. [S.l.]: [s.n.]. 2015. p. 727-732.

OREN, E. E. A. What are semantic annotations. [S.l.]. 2006.

PARK, S.; SHOEMARK, P.; MORENCY, L.-P. Toward crowdsourcing micro-level behavior annotations: the challenges of interface, training, and generalization. Proceedings of the 19th international conference on Intelligent User Interfaces. [S.l.]: [s.n.]. 2014. p. 37-46.

PAULIKS, R.; TRETJAKS, K.; BELAHS, K. A survey on some measurement methods for subjective video quality assessment. Computer and Information Technology (WCCIT), 2013 World Congress on. [S.l.]: [s.n.]. 2013. p. 1-6.

PEDERSEN, J. E. A. Conceptual foundations of crowdsourcing: A review of IS research. 46th Hawaii International Conference on IS research. Hawaii : [s.n.]. 2013.

PINTO, J. P.; VIANA, P. TAG4VD: a game for collaborative video annotation. Proceedings of the 2013 ACM international workshop on Immersive media experiences. [S.l.]: [s.n.]. 2013. p. 25-28.

RAINER, B. et al. Is one second enough? Evaluating QoE for inter-destination multimedia synchronization using human computation and crowdsourcing. Quality of Multimedia Experience (QoMEX), 2015 Seventh International Workshop on. [S.l.]: [s.n.]. 2015. p. 1-6.

RAINER, B.; TIMMERER, C. A quality of experience model for adaptive media playout. Quality of Multimedia Experience (QoMEX), 2014 Sixth International Workshop on. [S.l.]: [s.n.]. 2014. p. 177-182.

RAINER, B.; TIMMERER, C. Quality of experience of web-based adaptive http streaming clients in real-world environments using crowdsourcing. Proceedings of the 2014 Workshop on Design, Quality and Deployment of Adaptive Video Streaming. [S.l.]: [s.n.]. 2014. p. 19-24.

RAZAVIAN, M.; LAGO, P. A systematic literature review on SOA migration. Journal of Software: Evolution and Process, v. 27, p. 337-372, 2015.

REDI, M. et al. 6 seconds of sound and vision: Creativity in micro-videos. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.]: [s.n.]. 2014. p. 4272-4279.

RIBEIRO, F. E. A. Crowdmos: An approach for crowdsourcing mean opinion score studies. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.]: [s.n.]. 2011.

RIEK, L. D.; O'CONNOR, M. F.; ROBINSON, P. Guess what? a game for affective annotation of video using crowd sourcing. In: _____ Affective computing and intelligent interaction. [S.l.]: Springer, 2011. p. 277-285.

ROHWER, P. A note on human computation limits. Proceedings of the ACM SIGKDD Workshop on Human Computation, 2010.

SANCHEZ-CORTES, D. et al. In the Mood for Vlog: Multimodal Inference in Conversational Social Video. ACM Transactions on Interactive Intelligent Systems (TiiS), v. 5, p. 9, 2015.

SANTOS-NETO, E. et al. Towards Boosting Video Popularity via Tag Selection. SoMuS@ ICMR. [S.l.]: [s.n.]. 2014.

SCEKIC, O. C. D. A. S. D. Simulation-based modeling and evaluation of incentive schemes in crowdsourcing environments. OTM Confederated International Conferences" On the Move to Meaningful Internet Systems". Berlin: [s.n.]. 2013.

SCHWEIGER ET AL., F. Fully automatic and frame-accurate video synchronization using bitrate sequences. IEEE Transactions on Multimedia, 2013.

SCHWEIGER, F. TU München - Lehrstuhl für Medientechnik. Available: http://www.lmt.ei.tum.de/florian/sync/#data. [S.l.]: [s.n.]. 2012.

SEGUNDO, R. M. C. . M. N. D. A. A. C. A. S. S. CrowdSync: User generated videos synchronization using crowdsourcing. [S.l.]: [s.n.]. 2017~~~ çipi6m3.

SEGUNDO, R. M. C. et al. Systematic Review on Crowdsourced Videos. SUBMITTED TO PUBLICATION. [S.l.]: [s.n.]. 2016.

SEGUNDO, R. M. C.; DE AMORIM, M. N.; SANTOS, C. A. S. Crowdsourcing & Multimedia. 22nd Brazilian Symposium on Multimedia and the Web - Webmedia '16. [S.l.]: [s.n.]. 2016.

SEGUNDO, R. M. C.; SANTOS, C. A. S.; DE AMORIM, M. N. Systematic Review on Crowdsourced Videos. SUBMITTED TO PUBLICATION. [S.l.]: [s.n.]. 2016.

SEGUNDO, R.; DE AMORIM, M.; SANTOS, C. LiveSync: a Tool for Real Time Video Streaming Synchronization from Independent Sources. Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia). [S.l.]: [s.n.]. 2016.

SEGUNDO, R.; SANTOS, C. Remote Temporal Couplers for Multiple Content Synchronization. IEEE ICCIT. [S.l.]: [s.n.]. 2015. p. 532-539.

SHAHID, M. et al. Crowdsourcing based subjective quality assessment of adaptive video streaming. Quality of Multimedia Experience (QoMEX), 2014 Sixth International Workshop on. [S.l.]: [s.n.]. 2014. p. 53-54.

SHRESTHA, P. et al. Automatic mashup generation from multiple-camera concert recordings. Proceedings of the 18th ACM international conference on Multimedia, 2010. 541-550.

SIMON, H. A. Invariants of human behavior. Annual review of psychology, 1990. 1-20.

SIVAKORN, S. J. P. A. A. D. K. I'm not a human: Breaking the Google reCAPTCHA. [S.l.]. 2016.

SPIRO, I. Motion chain: a webcam game for crowdsourcing gesture collection. CHI'12 Extended Abstracts on Human Factors in Computing Systems. [S.l.]: [s.n.]. 2012. p. 1345-1350.

STEINER, T. et al. Crowdsourcing event detection in YouTube video. 10th International Semantic Web Conference (ISWC 2011); 1st Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web. [S.l.]: [s.n.]. 2011. p. 58-67.

STEINMETZ, R.; MEYERS, T. Multimedia synchronization techniques: experiences based on different system structures. Proceedings of Multimedia'92-4th IEEE COMSOC International Workshop on Multimedia Communications, Monterey, California. [S.l.]: [s.n.]. 1992. p. 306-314.

STEPHEN ROBERTSON, M. V. I. W. Rethinking the ESP game. CHI'09 Extended Abstracts on Human Factors in Computing Systems, 2009.

SU, H. J. D. A. L. F.-F. Crowdsourcing annotations for visual object detection. Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence. [S.l.]: [s.n.]. 2012.

SU, K. et al. Making a scene: alignment of complete sets of clips based on pairwise audio match. ACM International Conference on Multimedia Retrieval. [S.l.]: [s.n.]. 2012. p. 26.

SULSER, F.; GIANGRECO, I.; SCHULDT, H. Crowd-based semantic event detection and video annotation for sports videos. Proceedings of the 2014 International ACM Workshop on Crowdsourcing for Multimedia. [S.l.]: [s.n.]. 2014. p. 63-68.

TAG, B. et al. Collaborative storyboarding through democratization of content production. Proceedings of the 11th Conference on Advances in Computer Entertainment Technology. [S.l.]: [s.n.]. 2014. p. 40.

TAHBOUB, K. et al. An intelligent crowdsourcing system for forensic analysis of surveillance video. IS\&T/SPIE Electronic Imaging. [S.l.]: [s.n.]. 2015. p. 94070I--94070I.

TANG, A.; BORING, S. EpicPlay: crowd-sourcing sports video highlights. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. [S.l.]: [s.n.]. 2012. p. 1569-1572.

USTALOV, D. A. Y. K. Add-Remove-confirm: Crowdsourcing Synset Cleansing. Application of Information and Communication Technologies (AICT), 2015 9th International Conference on. [S.l.]: [s.n.]. 2015.

VASUDEVAN, V. et al. Is Twitter a good enough social sensor for sports TV? Pervasive Computing and Communications Workshops (PERCOM Workshops), 2013 IEEE International Conference on. [S.l.]: [s.n.]. 2013. p. 181-186.

VENKATAGIRI, S. P. et al. On Demand Retrieval of Crowdsourced Mobile Video. Sensors Journal, IEEE, v. 15, p. 2632-2642, 2015.

VILMOS ZSOMBORI, M. F. R. L. G. M. F. U. P. C. I. K. R. C. Automatic generation of video narratives from shared UGC. Proceedings of the 22nd ACM conference on Hypertext and hypermedia (HT '11). New York: [s.n.]. 2011. p. 325-334.

VLIEGENDHART, R. et al. How do we deep-link?: leveraging user-contributed time-links for non-linear video access. Proceedings of the 21st ACM international conference on Multimedia. [S.l.]: [s.n.]. 2013. p. 517-520.

VON AHN, L. et al. recaptcha: Human-based character recognition via web security measures. Science, v. 321, p. 1465-1468, 2008.

WANG, G. et al. Active key frame selection for 3D model reconstruction from crowdsourced geo-tagged videos. Multimedia and Expo (ICME), 2014 IEEE International Conference on. [S.l.]: [s.n.]. 2014. p. 1-6.

WANG, J. E. A. Crowder: Crowdsourcing entity resolution. Proceedings of the VLDB Endowment, 2012. 1483-1494.

WANG, O. et al. Videosnapping: Interactive synchronization of multiple videos. ACM Transactions on Graphics (TOG), v. 33, p. 77, 2014.

WANG, X. A. Q. W. Video Synchronization with Trajectory Pulse. Chinese Conference on Intelligent Visual Surveillance. Singapore: [s.n.]. 2016.

WILK, S.; EFFELSBERG, W. The influence of camera shakes, harmful occlusions and camera misalignment on the perceived quality in user generated video. Multimedia and Expo (ICME), 2014 IEEE International Conference on. [S.l.]: [s.n.]. 2014. p. 1-6.

WILK, S.; KOPF, S.; EFFELSBERG, W. Video composition by the crowd: a system to compose user-generated videos in near real-time. Proceedings of the 6th ACM Multimedia Systems Conference. [S.l.]: [s.n.]. 2015. p. 13-24.

WU, B. et al. Crowdsourced time-sync video tagging using temporal and personalized topic modeling. Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. [S.l.]: [s.n.]. 2014. p. 721-730.

WU, S.-Y.; THAWONMAS, R.; CHEN, K.-T. Video summarization via crowdsourcing. CHI'11 Extended Abstracts on Human Factors in Computing Systems. [S.l.]: [s.n.]. 2011. p. 1531-1536.

XU, P.; LARSON, M. Users Tagging Visual Moments: Timed Tags in Social Video. Proceedings of the 2014 International ACM Workshop on Crowdsourcing for Multimedia. [S.l.]: [s.n.]. 2014. p. 57-62.

YANG, J. E. A. Modeling Task Complexity in Crowdsourcing. Proceedings of The Fourth AAAI Conference on Human Computation and Crowdsourcing, 2016. 249-258.

YU ET AL., J. Understanding mashup development. Internet Computing, IEEE, 2008.

YU, H. et al. Challenges and opportunities for trust management in crowdsourcing. IEEE/WIC/ACM WI-IAT. [S.l.]: [s.n.]. 2012. p. 486-493.

ZEGARRA RODRIGUEZ, D.; LOPES ROSA, R.; BRESSAN, G. No-reference video quality metric for streaming service using DASH standard. Consumer Electronics (ICCE), 2015 IEEE International Conference on. [S.l.]: [s.n.]. 2015. p. 106-107.

ZHANG, C.; LIU, J. On crowdsourced interactive live streaming: a Twitch. tv-based measurement study. Proceedings of the 25th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video. [S.l.]: [s.n.]. 2015. p. 55-60.

ZHANG, L. et al. An Automatic Three-Dimensional Scene Reconstruction System Using Crowdsourced Geo-Tagged Videos. Industrial Electronics, IEEE Transactions on, v. 62, p. 5738-5746, 2015.

ZHAO, L. G. S. A. R. S. Robust Active Learning Using Crowdsourced Annotations for Activity Recognition. Human Computation, 2011.

ZHOU, Y. et al. Analyzing streaming performance in crowdsourcing-based video service systems. The 21st IEEE International Workshop on Local and Metropolitan Area Networks. [S.l.]: [s.n.]. April 2015. p. 1-6.