同世界放送システムのための映像収集木構築手法の検討

牧田 航輝[1]　川上 朋也[1]　松本 哲[2]　義久 智樹[2]　寺西 裕一[3,2]　下條 真司[2]

概要：我々は，これまで「同世界放送」と呼ぶライブ放送システムを検討してきた．同世界放送では，多数の遠隔地で撮影された映像を低遅延にリアルタイムで合成する必要があり，映像の収集と合成方法が重要となる．特定のサーバに集約させるとスケーラビリティに問題があるため，中継ノード上で合成しつつ収集する手法を考案してきたが，そのための収集木として 2 分木や 4 分木といった限定的な構造しか対象としてこなかった．そのため，本来考慮すべき各ノードの処理性能やネットワーク帯域などが考慮されていないという問題があった．本稿では，これまで検討してきた 2 分木等に限定されない収集木を動的に構築する手法について検討する．また，収集木をリアルタイムに決定するために，収集木の決定にかかる計算時間を削減する手法についても検討する．

1. はじめに

2019 年末に発生した新型コロナウイルス感染症（COVID-19）の影響により，テレワークやオンライン授業など，リアルタイム映像配信サービスの需要が高まっている．多くの映像配信サービスでは，複数の遠隔地で撮影された映像を，画面を分割することによって各映像を 1 つの画面内に表示する方法がとられている．画面を分割することなく撮影対象を同じ空間内に配置することができれば，高いエンターテインメント性を生み出したり，テレワークやオンライン授業の際に生じる違和感や不快感を軽減したりする効果が期待できる．

そこで，我々は「同世界放送」と呼ぶライブ放送システムを検討してきた [1–3]．同世界放送とは，多数の遠隔地で撮影された撮影体操が，まるで同じ空間に存在するかのようなライブ放送のことを言う．図 1 は，同世界放送で生成される合成映像の例である．特に，同世界放送では多数のリアルタイム映像をインターネット経由で収集し，低遅延に映像の合成と視聴者への配信を行う必要がある．しかし，特定のサーバに集約させ処理を行うと「撮影者の増加に伴い，合成の処理の負荷が増大し，処理の完了までの遅延が大きくなる」という問題が生じる．例えば，Microsoft Teams の Together mode [4] では最大 49 人までの参加者の映像が共通の背景に自動的に配置されるが，参加者の映像を集中型で収集合成するモデルでは処理サーバに高い処理性能が要求され，スケーラビリティにも限界がある．そこで，我々は中継ノード上で合成処理を分散し，低遅延に合成することを検討してきた．本稿では，従来より効果的な映像の合成手法を検討する．

図 1　同世界放送の例

2. 同世界放送システム

同世界放送の構成を図 2 に示す．同世界放送システムでは，撮影者からリアルタイム映像をインターネット経由で収集し，元映像の背景除去や複数映像の合成などの処理を行うことで同世界映像を生成する．また，特定のノードに処理負荷と通信負荷が集中し，合成が完了するまでの遅延が大きくなることを避けるため，中継ノード上で複数の映像を合成し，合成後の映像のみを次のノードに転送する方法をとる．合成の例を図 3 に示す．図の例では，7 つの映像を合成することを考える．A G は，それぞれが自身が撮

[1] 福井大学
University of Fukui
[2] 大阪大学
Osaka University
[3] 情報通信研究機構
National Institute of Information and Communications Technology

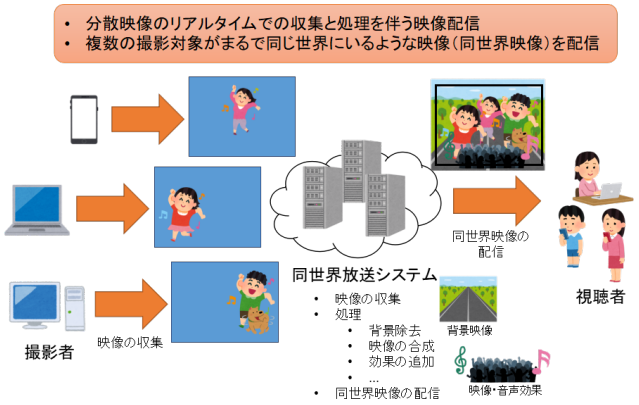

図 2　同世界放送の構成

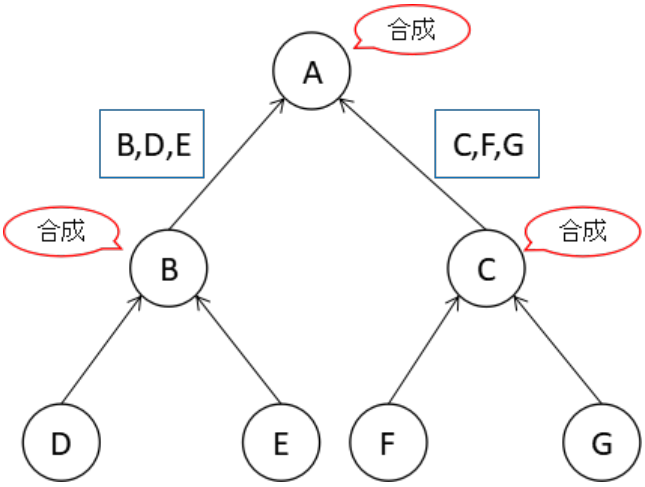

図 3　映像を合成しつつ収集する例

影した映像を持っているものとする．まず，B，D，E に注目すると，D，E からは自身の映像を B へと転送し，B では受け取った D，E の映像と自身の映像を合成した 1 つの映像を生成する．そして，その生成した映像を次の A に転送する．C，F，G でも同様の処理を行い，合成映像を A に転送する．そして，A で 7 つ全ての映像を合成するという流れになっている．このように，中継ノード上で複数の映像を合成し，合成後の映像のみを次のノードへ転送することで，特定ノードに対する通信負荷の削減や合成処理の分散を行える．

3.　映像の合成方法

3.1　これまでの合成方法

文献 [3] では，集中合成と分散合成という 2 つの映像の収集・合成方法を評価した．集中合成とは，問題点として挙げたように特定のノードに映像を集め合成処理を行うものを指す．一方で分散合成とは，前章で述べたように複数ノードで映像を合成しつつ収集するものを指す．分散合成として，デイジーチェーン，2 分木，4 分木の 3 パターンを構成し，集中合成を加えた 4 つの合成方法における合成の遅延時間を計測し評価した．評価の結果，特に合成数が多い場合，集中合成よりも 2 分木，4 分木のような収集木を構成することで遅延時間を短くすることができることを確認した．しかし，ここで問題となるのは分散合成における収集木の構成方法である．文献 [3] では，各ノードが撮影・合成した映像の転送先をあらかじめ指定し，決められた 2 分木や 4 分木を構成した．映像合成の処理負荷の違いによって適した合成方法が異なるという評価結果も得られており，一概にどの合成方法が良いと言うことは難しい．そのため，収集木を 2 分木等に固定することなく，各ノードの処理性能やネットワーク帯域を考慮し動的に収集木を構成することが望ましい．そこで，本稿では動的な収集木の構築手法を検討する．

3.2　検討する収集木の構築手法

まず，前提として 1 台の配信サーバに全ての映像が収集され，合成した後に配信することを想定する．つまり，配信サーバを根に持つ木を構成することを考える．配信サーバは，転送されてきた映像を合成し，ある一定のフレームレートで配信する．すなわち，フレーム間隔内に映像を受信し合成することができれば合成・配信し，フレーム間隔内に新しい映像が受信されなければ合成・配信は行わないということになる．この時，フレーム間隔内に中継ノードを介して配信サーバまで転送される映像と，中継ノード上の処理時間が大きくなったり配信サーバまでのホップ数が大きくなったりすることにより，フレーム間隔内に転送されない映像が混在していることが考えられる．フレーム間隔内に受信されない映像は更新されないため，最終的な同世界映像で止まったように見えることになる．つまり，最新の合成映像を配信し続けるためには，フレーム間隔内により多くの撮影ノードから映像を受信することが重要となる．ここで，配信サーバがフレーム間隔内に受信できる撮影ノードの数を UN，同世界放送に参加している撮影ノードの数を N とし，更新率 U を

$$U = \frac{UN}{N} \tag{1}$$

とする．この更新率を高くすることによって，より多くの映像が更新され臨場感のある同世界映像を得ることができることになる．

以上より，臨場感のある同世界映像の生成のため，更新率を上げることが目的となる．そのためには，フレーム間隔内に受信できる映像数を増やすことができればよい．フレーム間隔内に受信できるか否かは，各ノードで映像の合成にかかる時間とノード間の映像の転送にかかる時間を計算することによって求める．結局，考えられる様々な収集木に対して更新率を計算し，更新率ができる限り 1 に近づくような収集木を決定することとなる．

3.3 更新率の計算

撮影ノード n において，合成すべきフレームが増える毎に増加する時間を FP_n，フレーム数に関わらずかかるオフセット時間を IP_n とし，f 個のフレームを合成するのにかかる時間 $P_n(f)$ を

$$P_n(f) = f \times FP_n + IP_n \tag{2}$$

と定める．また，撮影ノード m の平均フレームデータサイズを FS_m，撮影ノード n, m 間の通信帯域を $B_{n,m}$ とし，f 個の撮影ノードのフレームを合成する撮影ノード n が m から 1 フレームを受信するのにかかる時間 $R_{n,m}(f)$ を

$$R_{n,m}(f) = \frac{FS_m}{B_{n,m}(f)} \tag{3}$$

と定める．ただし，撮影ノード i の入帯域を I_i，出帯域を O_i とし，

$$B_{n,m}(f) = \begin{cases} \frac{I_n}{f} & (\frac{I_n}{f} < O_m) \\ O_m & (\frac{I_n}{f} \geqq O_m) \end{cases} \tag{4}$$

とする．以上を用い，撮影ノード n から根（配信サーバ）への経路に含まれる撮影ノードの集合を RN_n，撮影ノード n の子の数を C_n，配信フレームレートを f とし，

$$UN = \underset{n \in N}{count} \left\{ n \middle| \sum_{k \in RN_n} (P_k(C_k) + R_{parent(k),k}(C_k)) < \frac{1}{f} \right\} \tag{5}$$

と求めることができ，更新率が決定される．

ところで，同世界放送に参加するノード数が N のとき，ある撮影ノード n の送信先候補数は，自身を除く撮影ノード $(N-1)$ 個と配信サーバ 1 個を含めて N 個ある．よって，考えられる収集木は N^N 通り存在する．前述の更新率の計算は，決定されたある 1 種の収集木に対して行うことになり，更新率が 1 となれば全ての映像が決められたフレームレートで更新されることとなり目的が達成されるので，そこで打ち切れば良い．しかし，更新率が 1 未満の場合，考えられる別の収集木に対して再度更新率の計算を行う必要がある．よって，総当たりで各収集木の更新率を計算すると最悪の場合 N^N 通りの収集木に対して更新率を計算する必要があり，更新率の高い収集木をリアルタイムで見つけることが困難となる．

3.4 計算量を削減する方法

同世界放送において，「各映像が重なり合う場合がある」という性質を用いて計算量を削減することを考える．図 4 は同世界放送における映像合成の例であり，1〜8 は参加者の映像を示している．図の例では，映像 1 は映像 2,3,4 の背面にあり，映像 6 は映像 3,7 の前面にあるというように，映像の重なり合いが生じている．このように遮蔽が発生することを考慮し，遮蔽グラフを生成する．遮蔽グラフは，

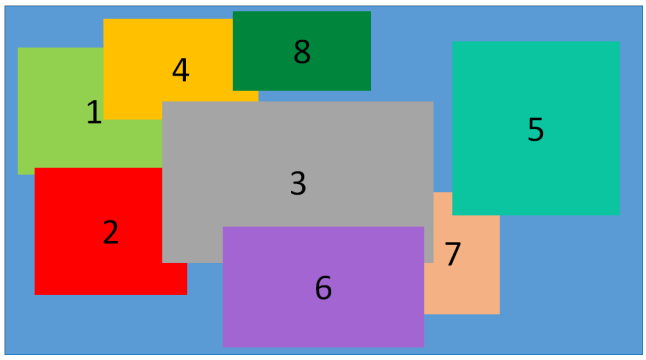

図 4 映像合成の例

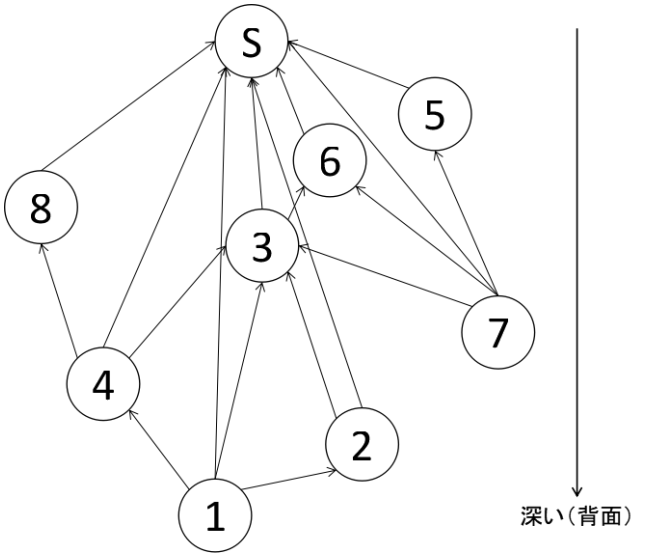

図 5 図 4 から生成される遮蔽グラフ

各映像の深度情報をもとに生成する．ある映像に注目した時に，その映像と重なっている映像について，前面にある映像には送信し，背面にある映像からは受信するように構成する．そうすることで，「背面から順に合成していく」という自然な流れを保ちつつ，送信先の候補を削減することができる．ただし，配信サーバは最前面にあるとし，全ての撮影ノードから受信できるようにする．図 4 の合成例から遮蔽グラフを生成すると図 5 のようになる．S は配信サーバを示しており，1〜8 は図 4 の映像を撮影するノードを示している．このような遮蔽グラフを生成するアルゴリズムを Algorithm1 に示す．なお，Algorithm1 では重なりや重なり順の判定に全ての映像の合成位置や深度を知る必要があるので，全てのノードは映像の合成位置や大きさ，深度を配信サーバと共有するものとする．

3.5 収集木の決定法

遮蔽グラフでは一般に，各ノードから送信先の候補として複数のエッジが作られることになるが，この中から 1 つのエッジのみを選択することによって 1 種の収集木が決定される．しかし，このようにして決定された収集木に従って，中継ノード上で受信した映像を全て合成すれば良いか

Algorithm 1 遮蔽グラフの生成

```
n:新しく追加するノード
N = {1, 2, ...} //n を除く撮影ノード集合
for i ∈ N do
  if n と i が重なっている then
    if n が i の背面 then
      n から i に送信するエッジを張る
    else
      i から n に送信するエッジを張る
    end if
  end if
end for
```

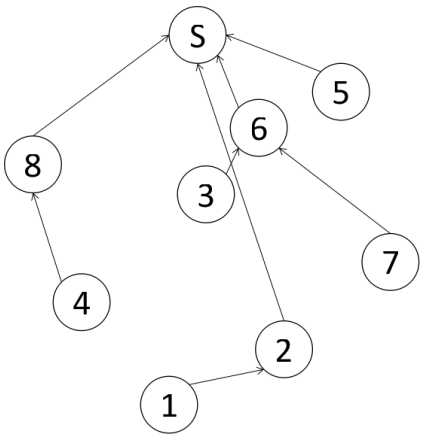

図 6 図 5 から決定した収集木の例（1）

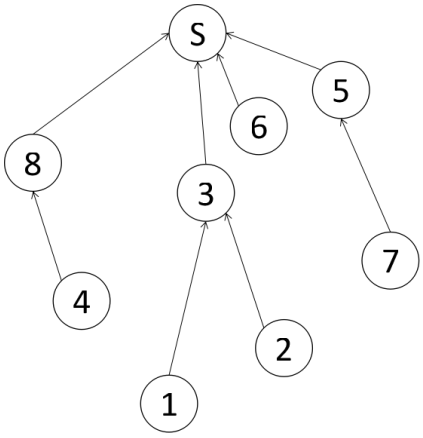

図 7 図 5 から決定した収集木の例（2）

というとそうとは限らない．例として，図 5 の遮蔽グラフから図 6 の収集木を決定した場合と，図 7 の収集木を決定した場合を考える．以下では，背面の映像 a と前面の映像 b を合成した映像を (a,b) のように表記する．図 6 では，(1,2)，(4,8)，(7,3,6)，(5) と合成した映像を S に送信し，S では，((1,2)，(4,8)，(7,3,6)，(5)) と合成すれば図 4 の合成映像が得られる．しかし，図 7 では，(1,2,3)，(4,8)，(7,5)，(6) と合成した映像が S に送信されることになるが，S でこれらの映像をどのような順で合成しようとも図 4 のようにはならない．例えば，((7,5)，(1,2,3)，(4,8)，(6)) と合成した場合には，本来映像 4 は映像 3 の背面に

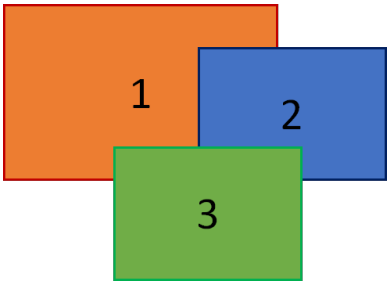

図 8 問題が生じ得る例

なければならないにもかかわらず，映像 3 の前面に合成されてしまう．そこで，合成順を入れ替えて ((7,5)，(4,8)，(1,2,3)，(6)) とした場合，映像 4 の背面になくてはならない映像 1 が映像 4 の前面に合成されてしまう．このように，中継ノード上で受信した映像を全て合成すると，S でどのように合成しても正しく合成することができない場合がある．

よって，遮蔽グラフから収集木を決定するに当たり，選択してはならない収集木が存在することに注意する必要がある．合成に注意する必要があるのは，ある映像が他の映像の中間に存在するような場合であり，図 8 がその例である．図 8 において，映像 2 は映像 1 と 3 の間に挟まるような形になっている．映像 2 を合成する前に映像 1 と 3 を合成してしまうと，映像 2 をその間に差し込むことができなくなる．このような場合には，3 つとも合成せずに次のノードへ送るか，1 と 2 または 2 と 3 を先に合成する必要がある．以上のような合成順を考慮した上で収集木を決定するアルゴリズムを Algorithm2 に示す．遮蔽グラフに対し Algorithm2 を適用することで 1 つの収集木が生成される．図 9 は，図 5 に Algorithm2 を適用した結果得られる収集木の例である．

このようにして生成された収集木に対して更新率の計算を行う．更新率が 1 でない場合にはより良い収集木が存在する可能性があるので，別の収集木を検討する必要がある．Algorithm2 は，初めに「S から最も深いノードまで，深度さが小さい方を選択していく」という操作によって，なるべく合成しつつ収集する中心となる経路を選択し，残りのノードをその経路に合わせていく方針をとっている．そのため，ホップ数が大きくなりがちであるので，フレームの送受信時間が大きくなり更新率が低下しやすい．しかし，配信サーバへの集中を避けられるので，配信サーバの処理性能が良くなかったり帯域が小さかったりする場合には有効であると言える．Algorithm2 で生成された収集木の更新率が悪い場合には，最初に S から最も近いノードを選択するのではなく，「S から 2 番目に近いノードを選択し，2 番目に近いノードは直接 S に送信する」という操作をすることで，ホップ数を減らすことができる．さらにホップ数を減らしたい場合には，S から 3 番目に近いノード，4 番

Algorithm 2 収集木の決定

```
S から最も深いノードまで，深度差が小さい方を選択していく
未決定ノードの内，送信先候補が S だけのノードは S に送信
node_list ← 未決定ノードを浅い順に並べた集合
while node_list が空でない do
  for i ∈ node_list do
    parents ← i から S までに含まれる浅い順に並べたノード集合
    children ← i から一番深いノードまでに含まれるノード集合
    if parents に受信元決定済みノードがある ∧ children に送信先決定済みノードがある then
      child ← children の内 i に最も近いノード
      for j ∈ parents do
        if 決定済みの経路上で child → j の経路が存在する then
          i は j に送る
          node_list から i を除外
        end if
      end for
    end if
  end for
  deep ← 未決定ノードの内最も深いノード
  if deep の親ノードに受信元が決定しているノードが 1 つある then
    そのノードに送る
  else if deep の親ノードに受信元が決定しているノードが 2 つ以上ある then
    deep の送信候補先とその送信元ノードの送信候補先がより多く一致しているノードに送る
  else
    一番近いノードに送る
  end if
  node_list から deep を除外
end while
```

目に近いノード... というように選択することで S に直接送信するノード数を増やせばよい．

しかし，このアルゴリズムで生成される収集木の数は限られており，全ての候補を計算したとしても良い更新率が得られない可能性もある．一方で，Algorithm1 で送信先候補を絞った後に Algorithm2 で合成順を考慮した収集木のみを生成しているので，無駄がなく計算量を小さくできており，更新率と計算量のトレードオフになっていると言える．

4. まとめ

本稿では，同世界放送における映像の収集と合成について，動的に収集木を構築する手法を検討した．特に，同世界放送で生じ得る重なり合いに注目し収集木の候補数の削減や重なり順を満たしつつ収集木を生成する方法を考案したが，まだまだ改善の余地がある．例えば，Algorithm1 では背面から前面へと映像を転送するように遮蔽グラフを生成したが，この方法は全く重なり合っていない部分同士で合成することによってより更新率を高められる可能性を排除してしまっている．また，Algorithm2 の収集木の決定で

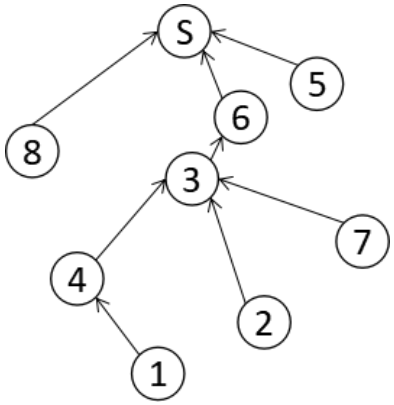

図 9 図 5 の遮蔽グラフに Algorithm2 を適用した結果得られる収集木

は，得られる収集木の数が少ないために更新率が芳しくない結果となる可能性がある．さらに映像の更新について，配信フレーム間隔内に受信されない映像は全く更新されず止まって見えるという問題もある．よって，今後はこれらの課題にも対応可能な更なる収集木構築手法の検討が必要となる．

謝辞 本研究の一部は G-7 奨学財団研究開発助成事業および福井大学研究育成経費，JSPS 科研費 18K11316 の助成による成果である．

参考文献

[1] 牧田航輝，川上朋也，松本　哲，義久智樹，寺西裕一，下條真司：リアルタイム映像の収集と合成を伴う同世界放送システムの検討，第 28 回情報処理学会マルチメディア通信と分散処理ワークショップ (DPSWS2020) 論文集, デモ発表，pp. 186–192 (2020).

[2] 牧田航輝，川上朋也，松本　哲，義久智樹，寺西裕一，下條真司：同世界放送：リアルタイム映像の収集と合成を伴う分散型インターネットライブ放送システム，マルチメディア，分散，協調とモバイル (DICOMO2021) シンポジウム，pp. 1568–1577 (2021).

[3] Makida, K., Kawakami, T., Matsumoto, S., Yoshihisa, T., Teranishi, Y. and Shimojo, S.: Same World Broadcasting: An Internet Broadcasting System for Real-Time Distributed Video Compositions, *Proceedings of the 9th IEEE International Workshop on Architecture, Design, Deployment and Management of Networks and Applications (ADMNET 2021)*, pp. 1423–1428 (2021).

[4] Lanier, J.: How to get the most from Together mode, `https://techcommunity.microsoft.com/t5/microsoft-teams-blog/how-to-get-the-most-from-together-mode/ba-p/1509496`.（参照 2021-09-21）.