

## 同期コミュニケーションへの途中参加を支援する ダイジェスト作成・提供方式

加藤 善大\* 石原 進† 酒井 三四郎† 水野 忠則†

筆者らは継続中の同期コミュニケーションへのスムーズな途中参加を実現するための方式として、ユーザ不在期間中の発言データを時間・サイズおよび重要度によって区分けしたブロックに分け、これらの組み合わせによって不在期間のダイジェストを作成・提供するブロック化ダイジェスト方式を提案している。本方式はユーザが同期コミュニケーションへ途中参加後すぐに発言できるようなダイジェストデータの提供のみならず、その再利用も目的としている。本稿ではブロック化ダイジェスト方式について述べ、同期・非同期電子会議システム DYNAMITE 上での実現と、同期コミュニケーションへの途中参加実験について報告する。

## The method for natural joining to the realtime communication

Yoshihiro Kato\*, Susumu Ishihara†, Sanshiro Sankai† and Tadanori Mizuno†

This paper describes the blocked digest method for asynchronous user who can join to a realtime communication in advance. A blocked digest method makes a digest which is made of several unit of digest-blocks. The digest-block is made by time or speech size, and distilled from an essential part of users communication (e.g. speech, text, applause and so on.). This method has a feature, that is not only a user can rejoin to the communication in advance, but also a system can reuse the digest-block to make an other new digest. We have implemented this system and merged into the DYNAMITE (that is a sync-async mobile teleconference system.), and we have evaluated it.

### 1 はじめに

近年めざましい発展を遂げているモバイルコンピューティング技術はユーザに行動の自由を与え、ユーザはいつでもどこでもコンピュータを使ったコミュニケーションが利用可能となりつつある。一方、行動の自由度とは裏腹にユーザの時間的・地理的な都合やその通信コストの高さから、モバイル環境下では同期コミュニケーションへの常時参加を制限される可能性が高い。

モバイル環境のみならず、固定通信環境においても遠隔地にいる多人数がネットワークを介して同期的なコミュニケーションを行う場合、ユーザの都合や通信環境の違いなどにより、すべてのユーザが同時刻にコミュニケーションへ参加することは難しい。

これに対し、コミュニケーションへの時間的に自由な参加・退出が可能である電子会議システムとして DYNAMITE (DYNAMIC Adaptive Mobile Teleconference system)[1]がある。DYNAMITE は会議途中に会議へ参加あるいは再参加するユーザ (非同期ユーザ) のためにすべての会議発言がサーバに蓄積され、マルチメディアデータベースとして発言の随時参照を可能とする。しかしながら、会議へ途中参加するよう

なユーザ (非同期形態) が過去の会話の流れを理解することなく会議に参加した場合、リアルタイム (同期形態) で進行している会議に参加・合流するタイミングを失いかねない。

そこで筆者らは蓄積された会議発言をその重要度や時間などのブロックに区分けし、これをダイジェストとしてユーザに提供することで、同期コミュニケーションに途中から参加するユーザがスムーズに会議に参加することを目的としたブロック化ダイジェスト方式を提案している [2]。

本稿では同期非同期電子会議システム DYNAMITE およびブロック化ダイジェスト方式とその実現について述べ、本方式を利用した会議への途中参加実験について報告する。

### 2 背景

#### 2.1 同期非同期電子会議システム DYNAMITE

DYNAMITE はインターネットを介して音声・映像・テキスト・ホワイトボード・拍手などによる多人数マルチメディアコミュニケーションを行うシステムである (図 1)。以下にその特徴を示す

- 通信路状態により通信形態 (同期、非同期) を切り替えることが可能
- 会議情報のダイジェストを提供することにより、ユーザの会議への途中参加や会議終了後の会議内容の参照を支援する

\* 静岡大学大学院理工学研究科  
Graduate School of Science and Engineering,  
Shizuoka University  
† 静岡大学情報学部  
Faculty of Information, Shizuoka University

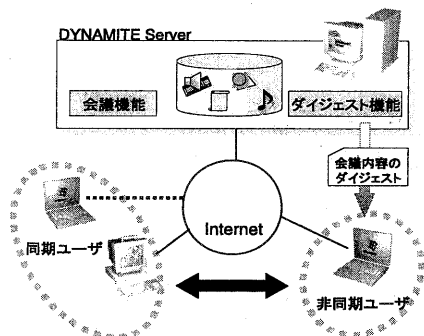


図1 同期・非同期電子会議システム DYNAMITE

- 音声発言の話速変換による発言参照の短時間化

本稿では特に二つ目のダイジェスト作成・提供による非同期支援に焦点を当てる。

## 2.2 ダイジェストによる非同期会議支援

DYNAMITE では全ての会議発言情報を蓄積保存している。非同期状態のユーザが同期会議に途中参加する際に、不在期間中の会議内容のダイジェストを作成・提供することで、ユーザは短時間でその内容を把握し会議へと合流することができる。この際ダイジェストは会議同様にマルチメディアデータによって提供される。これは文字や絵だけでは伝わらない会話の微妙なニュアンスなどを伝えるのに効果的であるからである。

しかし同期コミュニケーションへ途中参加するユーザにダイジェストを提供する場合、以下のような問題がある。

- ダイジェスト参照中に同期会議が進行する  
ユーザがダイジェスト参照している間にも会議が同時進行してしまい、ユーザはその間の会議内容を知ることが出来ない。このため、ダイジェストを参照して途中参加したとしても、参加直前の情報不足により、参加後すぐに発言できる状態であるとは限らない。
- ダイジェストデータの再利用性  
複数のユーザがダイジェストを要求した場合、たとえそれが同時刻における要求であっても、ユーザ毎にダイジェストを作成していたのではサーバに負荷がかかり、ダイジェスト結果を迅速に提供できないことも考えられる。特にダイジェスト作成処理が高度な場合は顕著である。

このように途中参加する非同期ユーザに対してダイジェストを提供することは、不在期間中の会議内容を迅速に把握できるという点で有効であるが、同時に途中参加に最も重要な途中参加時刻付近の会議内容を十分に理解できないという問題がある。

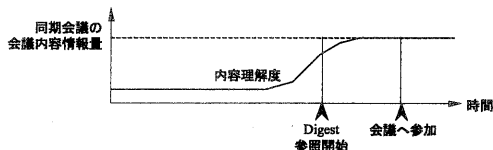


図2 スムーズな途中参加のための会議内容理解

## 3 ブロック化ダイジェスト方式

ブロック化ダイジェスト方式は前述のダイジェスト作成・提供における問題点を考慮し、次の2つを目的としている。

- 同期コミュニケーションへのスムーズな途中参加を考慮したダイジェスト作成・提供
- ダイジェストの再利用性

**同期コミュニケーションへのスムーズな途中参加** 本方式の目指すところは、ダイジェストを提供することにより、なるべく短時間でユーザ不在期間中の会議の流れを把握し、同期会議参加直後に会話に参加できる状態にすることである。

同期会議参加直後に会話に参加するためには、会議全体の流れに加え、参加直前の会話の内容を理解していることが重要と考える。そこで図2に示されるように、ユーザ不在期間中の会議内容の理解度が同期会議参加直前に向かって段階的に増加するようにダイジェストを作成する。

**ダイジェストの再利用性** 本方式では会議時間のある期間毎のブロックに分割する。ブロック中の発言にはその発言の重要度に対応した優先度を付与し、ブロック内をさらに優先度毎に分割し格納する。ユーザの到着時には、そのユーザの要求するダイジェスト参照量を、実際の再生時間を考慮しつつブロックを選択することでダイジェストを提供する。

これにより、あらかじめ作成されたダイジェストのブロックを組み合わせることによってダイジェストを提供できる。このため、同期会議参加時刻が異なる複数のユーザに対して、図2のような理解度曲線を与えるダイジェストを容易に提供可能である。

ブロック化ダイジェストの作成から提供までの流れを図3に示す。以後、本方式のブロック化によるダイジェスト作成および提供について述べる。

### 3.1 ブロック化によるダイジェストの作成

本方式では会議時間または発言量を元に、会議のある一定期間ごとにブロックとして分割する。このように分割されたブロックを区間ブロックと呼ぶ。また、区間ブロック内の発言に対し優先度の抽出を行い、その優先度に対応したブロックを作成する。これを優先度ブロックと呼ぶ。

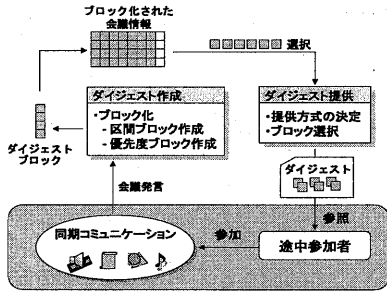


図3 ブロック化ダイジェスト方式の流れ

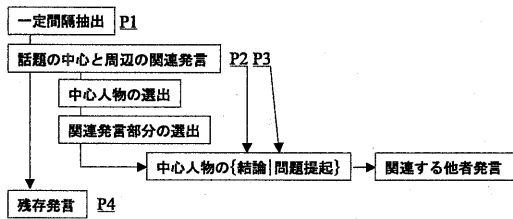


図4 優先度抽出の基本概念

### 3.1.1 区間ブロック分割

区間ブロックは会議発言のデータサイズを指標として分割する。ただし、発言が少なく、ある一定期間を経過してもブロックが作成されない場合には、指定のデータサイズに満たない場合でも一つのブロックとして分割する。

### 3.1.2 優先度ブロック分割

ブロック内に設定される優先度は1を最高として少なくとも4以上の優先度付けを行う。また各優先度に対し、システムまたはユーザがブロック中の全データサイズの何%を抽出するかを指定し、指定量の発言を抽出する。優先度の段階数および抽出割合は会議毎に指定可能であるが、基本的には4段階の優先度を付ける。

DYNAMITEではユーザの発言と同時に会議ログを取っており、動画、音声、テキスト、ホワイトボード、拍手などの発言情報をはじめとして、会議開始時間、ログイン時間、ユーザの形態、ユーザ情報、その他を格納している。発言の優先度抽出はこの会議ログを元に行われる。

次に各優先度における発言抽出の基本的な考え方を述べる。また図4にその概略を示す。

#### 優先度1

会議の内容を理解するというより意味の流れを把握することを重視し、最も選択される可能性のある優先度1では一定間隔で発言を抽出する方法をとる。これまでのダイジェスト参照実験により、ごく少量の発言抽出量で会議内容を理解するためには発言間の関連を考慮して抽出を行うよりも、一定間隔で抽出するほう

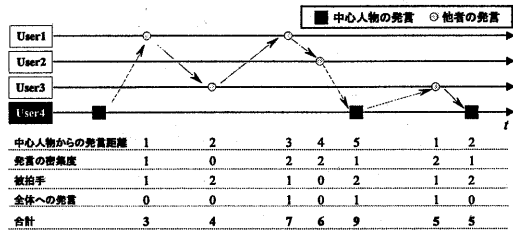


図5 中心人物の結論の抽出例

がユーザが会議内容の流れを判断するためには良いことがわかっている [3]。

### 優先度2 および3

優先度2, 3は優先度1における会議の流れの理解に加え、重要部分の内容理解を目的とする。そのための方針は区間ブロック中における会話の中心人物と他者とが活発に意見を交換している場所を優先的に抽出することとする。これは、通常の会議であれば、議長や発言の活発なユーザのようなコミュニケーションの中心となるユーザの発言を軸に会話を進行していくことが考えられるからである。

中心人物の選出は、それぞれのユーザの「発言時間」、「発言回数」、「被指名回数」、「被拍手回数」、「全てのユーザに対する発言数(広報発言)」を指標として、これらに重み付けを行い総合点の高いユーザを中心人物とする。

次に、中心人物を基点として他者との意見がやりとりされている場所を選出する。これはユーザ間の発言遷移や、中心人物が誰に対して発言しているかを調査することで判断できる。

話題の結論は会話中の決定事項と結びつく可能性が高い。そこで、一連の会話中で中心人物のある発言に注目した場合、その発言より以前の他者発言数、各発言の時間的密度、被拍手量、広報的発言を指標として総合点の高い中心人物の発言を話題の結論とみなし(図5)、これを優先度2とする。優先度3では優先度2と同様の手法を用い、中心人物による一連の会話の開始を問題提起的発言とみなし発言抽出を行う。

#### 優先度4

これまでの発言抽出では選択されなかった発言を選択する。

### 3.2 ダイジェストの提供

ダイジェストの提供は、優先度毎に分割されたブロックを途中参加ユーザの希望するダイジェスト抽出量だけ選択し、これをダイジェストとしてユーザに提供する。

しかしながら2.2で述べた通り、ユーザはダイジェストの参照中に同時進行する会議の内容を知ることが出来ない。したがってユーザはこの空白時間が長いほ

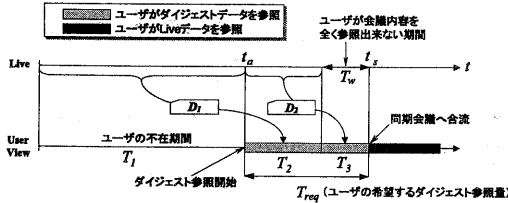


図 6 2段階によるダイジェストの提供

ど会議への途中参加に重要である直前の会議内容を失うことになる。そこで、本方式ではダイジェストの提供を図 6 に示すように 2 段階に分けることでユーザに対し会議へのスムーズな途中参加が出来るよう支援する。

次に 2 段階によるダイジェスト提供について述べる。

### 3.2.1 2 段階によるダイジェスト提供支援

ユーザの要求するダイジェスト参照時間を  $T_{req}$  とする。1 段階目ダイジェストの作成対象区間 (ユーザの不在期間) を  $T_1$  とし、 $T_1$  区間を対象としたダイジェストを  $D_1$  とする。このとき  $D_1$  を参照するのに必要な時間は  $T_2$  であり、同時に 2 段階目のダイジェスト作成対象区間が  $T_2$ 、その参照時間が  $T_3$  となる。また、ユーザがダイジェストを参照し始める時刻を  $t_a$ 、会議に参加できる時刻を  $t_s$  とすると、時刻  $t_a + T_2$  から  $t_a + T_2 + T_3$  までの間 ( $T_3$  区間)、ユーザは会議の内容を全く知ることができない。

本方式はユーザの要求するダイジェスト提供目標を満たす適切な  $T_2, T_3$  を決定し、これをダイジェスト提供目標量として優先度ブロックの選択を行いユーザに提供する。ブロックの選択は図 2 のような理解度を得るよう、図 7 に示す番号順に従い会議参加直前の会議情報を優先して選択する。

会議参加直前の情報欠損を最小限に押さえるために  $T_3$  を短くしたいが、同時に参加直前の情報をより多く知るために  $D_2$  を大きくしたい。  $T_3$  を短く設定することで空白時間を短くすることが出来るが、これは同時に  $D_2$  量が減り参加直前の会議情報を減少させる結果となる。このように空白時間の短縮と参加直前の会議情報の重視は相反特性を持つ。実装においてはこれらの適切な妥協点を用いて  $T_2$  および  $T_3$  を決定する必要がある。

そこで、ユーザが我慢できる限度の空白時間を仮に  $T_w$  とし、空白時間がこれよりも小さくなり、かつ  $D_2$  となるべく大きくなるように  $T_2, T_3$  を決定する。ただし、ユーザからのダイジェスト要求時間が空白限度時間  $T_w$  より小さい場合には、ユーザがダイジェスト参照中の空白時間を許容できるとして、1 段階のみのダイジェスト提供を行う。

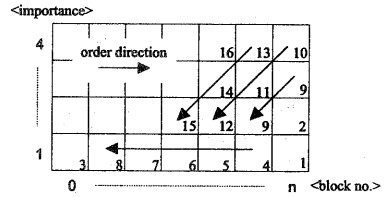


図 7 ブロック選択方針

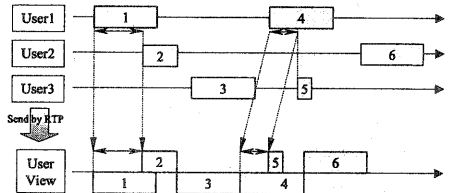


図 8 ダイジェスト発言の送信順序制御

### 3.2.2 最適な $T_2, T_3$ 時間の決定

$T_1$  期間中で実際に会話が行われていた時間の割合を発言存在率  $R (< 1)$ 、話速変換による発言短縮率を  $C (< 1)$ 、 $T_2$  期間中の全発言のうちダイジェストとして選択される発言の割合を  $d_2$  とする。  $T_2$  期間中に予想される会議発言時間は  $T_2 R$  であるから、ダイジェスト  $D_2$  の参照時間  $T_3$  は

$$T_3 = T_2 R C d_2$$

と計算できる。

$D_2$  を参照中に行われた発言のうち、ユーザがどれだけの発言を参照すべきかを定義し、その基準を満たす値を適切な  $T_3$  とする。本方式では  $T_2$  期間中に少なくとも半分以上の会話を知る必要があると仮定し  $d_2 = 0.5$  以上とする。これにより、

$$d_2 = \frac{1}{R \times C} \frac{T_3}{T_2} \geq 0.5$$

および

$$T_3 \leq T_w$$

を満たすような  $T_3$  値を適切な空白時間  $T_3$  の目標値とし、 $T_2 = T_{req} - T_3$  を  $T_2$  の目標値とする。

### 3.2.3 ダイジェストの送信

DYNAMITE では多人数同時発言が認められているため、2 人以上の発言が時間的に交差する状況がある (交差発言)。このような交差発言がダイジェストとして選択された場合、その交差状況を再現しつつ短時間で全発言を送信する必要がある。そこで図 8 に示すように、ダイジェスト中でも会話の重なりを再現し、かつ発言間の空白時間をゼロにするようにダイジェストの送信を行う。

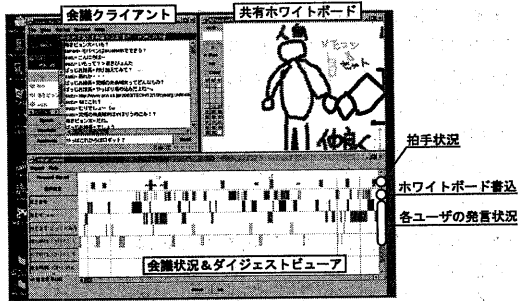


図 9 会議画面

## 4 実装と実験

### 4.1 実装

実際にブロック化ダイジェスト方式を DYNAMITE システム上で実現した。また、これに合わせ DYNAMITE システムを大幅に変更した。また、会議クライアントにダイジェスト参照用のビューア(会議進捗状況の確認や、単一発言の参照用を兼ねる)を組み込んだ。利用出来るメディアは音声・テキスト・ホワイトボード・拍手であり、実装上の問題からビデオの実装は見送った。音声・チャット・ホワイトボード・制御データは TCP、音声発言の送信は Unicast/RTP(DVI Codec 32Kbps) を利用した。実装言語は Java であり、マルチメディアライブラリとして Java Media Framework [4] を利用している。

### 4.2 実験と考察

ブロック化ダイジェスト方式を組み込んだ DYNAMITE システムで実際に会議を行った(図 9)。会議時間は約 1 時間程度で 2 名の途中参加者を用意し、会議中盤以降から実際にダイジェストを参照することで会議への途中参加をしてもらい、感想を聞いた。

(1) 会議条件 会議という名目上、調停役として 1 名の参加者を指名したが、各会議発言に発言権・書込権などの制限を設けず比較的自由に発言を行う雰囲気の中で会議を行った。なお、拍手は他者の発言に対し「重要である」「賛同できる」と判断した場合に実行してもらった(拍手ボタンをクリックすることで実行出来る)。途中参加者はほぼ同時刻に途中参加し、それぞれ 90 秒と 180 秒のダイジェストを要求してもらった。途中参加者 1 は 90 秒のダイジェストを要求するが、これはダイジェスト要求量が前述した空白時間 ( $T_w \leq 90$ ) の限度設定値内であるため、一段階のみのダイジェストを作成して提供する。また、途中参加者 2 は 180 秒のダイジェストを要求し、ダイジェスト要求量は限度設定値 90 秒以上であるため、2 段階のダイジェスト提供を行う。また、実際のダイジェストは音声発言のみを参照してもらった。

発言の優先度抽出の指標は前述の通りであり、その

表 1 会議条件

会議内容	究極の未来端末
会議時間	73 分
参加者	大学生、大学院生 6 名 (内 2 名が途中参加)
音声縮率	0.66 (1.5 倍速)
利用メディア	音声、テキストチャット、ホワイトボード(WB)、拍手
途中参加者 1	が会議開始から 46 分後に途中参加のため 90 秒のダイジェストを要求
途中参加者 2	が会議開始から 43 分後に途中参加のため 180 秒ダイジェストを要求

割合を表 3 に示す。優先度 1 は会議の流れを把握するには会議全体の 18% 程度の発言を参照する必要があるという報告 [3] から 20% とし、その他の割合は単純にほぼ 1/4 ずつ抽出するよう設定した。

(2) 会議結果 会議結果は表 2 のようになった。会話の内容は会議開始から 10 分までが「ペン型コンピュータ」について議論しており、10~25 分間は「体内埋め込み型」、25~30 分まで「ナノマシン」、35 分以降は「ロボット型」の端末の話題であった。最終的にロボット型が究極の未来端末であるという結論で会議が終了した。

会議の進行をつとめる User1 が大半の会話を行っており、には User1 が話題を他者に振る、または他者から話題を集める形態の会話となった。

優先度抽出の際に各区間ブロック期間中で中心人物と判断された回数はユーザ毎に表 2 のようになっている。User1 が最も多く中心人物と判断されているが、これは User1 の会話数、会話時間が他者よりも極端に多いからである。発言数、発言時間もに近い値の User2, 3 で User2 が中心人物として選択されている理由として、User2 が他者からの拍手をより多く受けていたことが考えられる。

表 2 会議結果

	発言数	発言時間	拍手数	WB 書込数	CHAT 発言数	中心人物として選択された数
User1(調停役)	114	1146	284	391	35	12
User2	51	357	309	75	45	3
User3	59	314	359	66	20	0
User4	38	245	304	169	19	0
途中参加者 1	17	97	0	0	11	0
途中参加者 2	13	136	1	15	0	0
総数	293	2296 秒	1296	702	145	-
サイズ	9185KB	-	45KB	229KB	11KB	-

表 3 優先度抽出方式と実験での平均抽出率

	抽出方式	目標抽出率 (%)	実際の平均抽出率 (%)
優先度 1	一定間隔抽出	20	22.4
優先度 2	中心人物(結論)	25	23.9
優先度 3	中心人物(問題提起)	25	23.3
優先度 4	残存発言	30	26.6

作成された区間ブロック数と 1 区間ブロックあたりの平均発言数は以下ようになった。約 5 分間ごとに 1 つの区間ブロックが作成された。

- 区間ブロック数: 15 (優先度ブロック数: 60)
- 1 区間ブロック当たりの平均発言数: 19.5

表 4 ダイジェスト要求結果

ダイジェスト要求量	途中参加者 1		途中参加者 2	
	90 秒	180 秒	153 秒	171 秒
D1 選択目標時間	90 秒	6ブロック選択 (12発言)	153 秒	7ブロック選択 (25発言)
実験中 D1 選択時間	97 秒		171 秒	
D2 選択目標時間	-		27 秒	3ブロック選択 (6発言)
実験中 D2 選択時間	-		29 秒	
要求から参加終了までの経過時間	97 秒 (ユーザ要求より 7 秒超過)		202 秒 (ユーザ要求より 22 秒超過)	

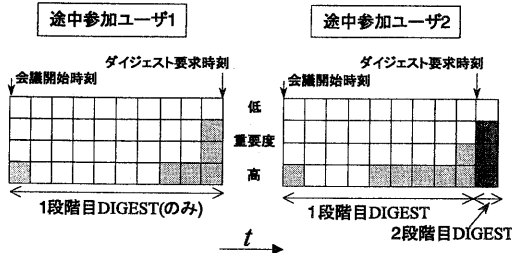


図 10 ダイジェスト要求時の選択ブロック

(3) 途中参加者へのダイジェスト提供 表 4, 図 10 に途中参加者 1, 2 のダイジェスト要求とブロック選択の状況を示す。図 10 から、スムーズな途中参加を考慮し、会議への参加直線付近の会議情報を重視してブロックを選択していることがわかる。結果としてユーザの要求より 7 秒多い 97 秒のダイジェストが選択されたが、これは 10% 程度の誤差であり許容範囲であると考えられる。

また、途中参加者 2 は会議開始から 43 分後に 180 秒のダイジェストを要求し、表 4 のようなダイジェスト選択目標時間を得た。途中参加者 2 のダイジェスト要求時刻までの総発言時間は 1265 秒、総会議時間は 2 555 秒、発言存在率は 0.49 であり、結果として要求より 22 秒多いダイジェスト参照量となった。これも要求量と大きく違っているわけではない。

(4) 途中参加者の感想 会議終了後に途中参加者に感想を聞いた。

ダイジェストの参照時間について ダイジェスト参照時間量について質問をしたところ、途中参加者 1, 2 両者ともに「特に気にならない」とのことであった。

しかし、180 秒のダイジェストを参照した途中参加者から「会議の要点を得ていたように思えるが、音声のみを 180 秒も聞き続けるのは精神的につらい」との意見を得た。今回のダイジェスト参照実験では音声発言のみ提供しており、各音声発言に関連するホワイトボード描画やチャット、拍手などは再現していない。このため変化の乏しい画面を見つめながら音声のみを聞くという精神的苦痛と、参照中の発言が会議中で占める位置が不明であること、また、発言に関連する他発言との関係が明確にならないことが参照者にダイジェストへの不信感を与えていると考えられる。

今後はダイジェストの音声発言に関連するホワイトボード書込や、拍手、チャットなどを複合的に利用することで、会議内容理解を深める工夫が必要である。

ダイジェストの参照内容について ダイジェストが内容的に問題無いかという質問をしたところ、特に 90 秒のダイジェスト参照者からは「内容はなかなか良い。会議の目的と現時点での話題が把握できた。会議参加がスムーズに行えた。」という感想を得た。これは本方式の参加直前の会議内容を重視するという特徴が現れた結果である。

発言の再生タイミング・話速変換の速度について 音声発言の連続参照という点について質問を行ったところ、「はい」「いいえ」「そう」などの極短時間の発言がとぎれる場合が多い」という意見を得た。電子会議クライアントでの音声発言はマウスをクリックしている間だけ音声を送信するというものである。このため短時間の発言になるほどクリック時間が短くなり、語尾がとぎれやすい原因となっており、発言インタフェースの改善をする必要がある。

再生タイミング、話速変換の速度については「タイミング・速度ともに問題なく会話の内容も十分わかる」と良好な感想を得た。

その他 本方式は会議をある期間のブロックごとに分割するが、話題の移り変わりでブロックを分割しているわけではない。このため今回の実験会議の前半部分のように比較的短時間で話題が移り変わる形態の会議では、区間ブロック分割のタイミングによっては隣接する区間ブロックの内容が全く異なる可能性がある。今後はブロック分割に意味要素を取り入れたり、ブロック間の関係を考慮する必要があると考える。

## 5 まとめ

ブロック化ダイジェスト方式を DYNAMITE 上に実現し、簡単な会議を行うことにより、電子会議という同期コミュニケーションへ実際に途中参加 p する実験を行った。途中参加者から比較的良好的な感想を得られたが、音声のみのダイジェスト参照では不十分であることがわかった。今後はさらに実験を重ねつつ、ダイジェストを音声を他の発言メディアと共に効果的に提示する方式について検討する。

### 参考文献

- [1] 太田賢, 山田善大, 奈良岡将英, 渡辺尚, 水野忠則, “モバイルコンピューティング環境における協調作業を支援する電子会議システム”, 情報処理学会論文誌, Vol. 39, No. 10, pp. 2879-2887, October 1998.
- [2] 加藤善大, 石原進, 酒井三四郎, 水野忠則, “非同期電子会議へのスムーズな再参加を支援するための一方式”, 情報処理学会高度道路交通システム研究グループ研究報告, 情処研報 Vol.99, No. ITS-3, pp. 57-64, 1999.
- [3] Y.Yamada, K.Ohta, and T.Mizuno, “Extracting and viewing information method for mobile teleconference system,” In proc of Asia Pacific Computer Human Interaction (APCHI'98), pp. 430-435 July 1998.
- [4] <http://java.sun.com/products/java-media/jmf/>