

SuperSQL における 3 次元仮想博物館の自動生成機構の実装

小谷美紗登[†] 五嶋 研人[†] 遠山元道^{††}

[†] 慶應義塾大学理工学部情報工学科 〒 223-8522 神奈川県横浜市港北区日吉 3-14-1

E-mail: †{kotani,goto}@db.ics.keio.ac.jp, ††toyama@ics.keio.ac.jp

あらまし SuperSQL は、独自のクエリを記述することによって関係データベースの出力結果を構造化し、多様なレイアウト表現を可能とする SQL の拡張言語である。また、仮想博物館とは博物館を 3 次元仮想空間で表現したものである。現在、データベースの取得結果は様々な表現がなされているが、ショッピングサイトの商品の画像など、一部の検索結果は 2 次元よりも 3 次元で表現する方がユーザの理解度が高まると考えられる。そこで、本論文では SuperSQL の質問文による関係データベースの取得結果から、様々なレイアウトで 3D オブジェクトを配置した 3 次元仮想博物館を自動生成するシステムを提案する。

キーワード VR, Unity, SuperSQL

1. はじめに

SuperSQL は、独自のクエリを記述することによって関係データベースの出力結果を構造化し、多様なレイアウト表現を可能とする SQL の拡張言語である。通常の SQL では、シンプルでフラットな表しか再現できないが、SuperSQL を用いることで様々な表を作成することができる。また、SuperSQL を用いることによって、従来の方法に比べてはるかに少ないコードで Web ページや phpなどを生成することができる。

現在、データベースの取得結果は様々な表現がなされているが、ショッピングサイトの商品の画像など、一部の検索結果は 2 次元よりも 3 次元で表現する方がユーザの理解度が高まると考えられる。そこで SuperSQL を拡張し、SuperSQL の質問文により関係データベースの取得結果から C# と XML のスクリプトファイルを自動的に生成し、それをゲームエンジンである Unity 内で使用することによって様々なレイアウトで 3D オブジェクトを配置した 3 次元仮想空間を生成するシステムを提案する。このシステムにより、Unity のオブジェクトの定義と簡単な SuperSQL の質問文の記述によって、容易に複雑な 3 次元仮想空間を作成することが可能となった。

以下、本稿の構成を示す。まず 2 章では SuperSQL の概要を述べ、3 章では関連技術、関連研究について、4 章では 3 次元仮想空間の生成について述べる。5 章では評価、検討について述べ、そして最後に 6 章でまとめについて述べる。

2. SuperSQL とは

この章では、SuperSQL について簡単に述べる。SuperSQL は関係データベースの出力結果を構造化し、多様なレイアウト表現を可能とする SQL の拡張言語であり、慶應義塾大学遠山研究室で開発されている [1][2]。そのクエリは SQL の SELECT 句を GENERATE <media> <TFE> の構文を持つ GENERATE 句で置き換えたものである。ここで <media> は出力媒体を示し、HTML, PDF などの指定ができる。また <TFE> はターゲットリストの拡張である Target Form Expression を

表し、結合子、反復子などのレイアウト指定演算子を持つ一種の式である。

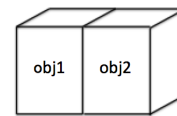
2.1 結合子

結合子はデータベースから得られたデータをどの方向 (次元) に結合するかを指定する演算子であり、以下の 3 種類がある。括弧内はクエリ中の演算子を示している。

- 水平結合子 (,)

データベースから得られたデータに沿って、3D オブジェクトを x 軸方向に配置する。

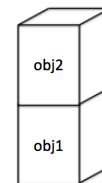
例: obj1, obj2



- 垂直結合子 (!)

データベースから得られたデータに沿って、3D オブジェクトを y 軸方向に配置する。

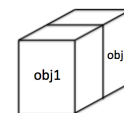
例: obj1! obj2



- 深度結合子 (%)

データベースから得られたデータに沿って、3D オブジェクトを z 軸方向に配置する。

例: obj1 % obj2



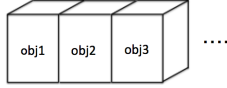
2.2 反復子

反復子は指定する方向に、データベースの値があるだけ繰り返し表示する。以下、その種類について述べる。

- 水平反復子 ([],)

データインスタンスがある限り、その属性のデータの 3D オブジェクトを x 軸方向に繰り返し配置する。

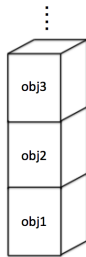
例：[Obj],



- 垂直反復子 ([]!)

データインスタンスがある限り、その属性のデータの 3D オブジェクトを y 軸方向に繰り返し配置する。

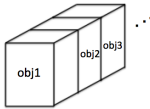
例：[Obj]!



- 深度反復子 ([]%)

データインスタンスがある限り、その属性のデータの 3D オブジェクトを z 軸方向に繰り返し配置する。

例：[Obj]%



2.3 複合反復子

ある方向 (次元) での反復の上限を指定し、これを越える場合に次に指定した方向 (次元) に伸長する。

[<TFE>] < 結合子 >< 回数 >< 結合子 >< 回数 >...< 結合子 >

例：[Obj],3!

3D オブジェクトを x 軸方向に 3 つ並べたものを、y 軸方向へ伸長していく。

例：[Obj],4%3!

3D オブジェクトを x 軸方向に 4 つ並べたものを、z 軸方向に 3 つ並べていき、まだ出力するオブジェクトがある場合はこの動作を y 軸方向へと伸長していく。

2.4 装飾子

- 表示する展示物名の変更

各展示物に表示する展示物名はデフォルトでは id となっているが、装飾子 (@) を用いて属性を指定することによって、表示する展示物名を変更することができる。

id@{name='表示名の属性'}

- 博物館の建物、展示台の変更

デフォルトで指定されているオブジェクト (博物館の建物、展示台) は変更可能となっており、使いたい博物館の建物や展示台のオブジェクトがあるフォルダを装飾子 (@) で指定することによって変更する。博物館の建物と展示台のテンプレートは、デフォルトのオブジェクトと同じスケールにしてあればユーザが作成したものも追加可能である。

<TFE>@{'装飾子の名称 = フォルダ名'}

博物館の建物を変更する場合は装飾子の名称に museum を指定し、展示台の場合は stand を指定する。複数指定するときは各々を “;” で区切る。

[color,[obj],!@{museum='Type1', stand='Type2'}]

3. 関連技術、関連研究

3.1 関連技術

このシステムでは仮想空間を表示するツールとして、幅広く使用されているゲームエンジンである Unity を使用している。Unity で開発したゲームは、Windows, Mac, iOS など、マルチプラットフォームに対応しており、ゲームのみならずバーチャル技術にも応用されている。また、Unity にはアセットストアという、ユーザが作成した 3D オブジェクトやスクリプトなどを公開、販売でき、また取得をすることもできるプラットフォームがあり、これによって豊富な機能拡張ができたり、クオリティの高い開発を行う事が可能となっている。

3.2 関連研究

Daniel Sacher らは、仮想博物館内でタスクに対して訪問者同士で協力することができるコラボレイティブ仮想博物館を考案し、そのツールチェーンを開発した。この研究は主に教育で応用することができると考えられている [5]。Chairi Kiourt らは DynaMus という、ユーザが簡単かつ自由に仮想博物館を作成することができる上に、Web 上にあるコンテンツで仮想博物館を生成することが可能である仮想博物館フレームワークを開発している。このフレームワークでは仮想博物館を表示するツールとして Unity を使用している [6]。A. Martina らは、文化機関が膨大な量の文化データベースを保持しており、それは専門家によって設計され管理されているが、非専門家がこのデータベースにアクセスすることが困難であることから、非専門家のために文化データベースへのアクセシビリティを改善するための新しい視覚的インタフェースとしてデータベースとバーチャルリアリティ技術の統合を提案している [7]。Ciro Donalek ら

は、科学的データの視覚化のための没入型仮想現実プラットフォームを開発しており、これによってデータスケールのジオメトリ、およびデータ内の関係のよりよい認識と、より直感的なデータの理解ができるようになった [8] .

4. 3次元仮想博物館の生成

この章では、3次元仮想空間を生成するシステムについて説明し、実際にこのシステムを使用した際のクエリと生成物について述べていく。

4.1 システムの概要

まず初めに博物館の展示物となる 3D オブジェクトは事前に用意し、それに関するデータをデータベースに格納する必要がある。SuperSQL のクエリを実行すると、そのデータに基づいて二つのスクリプトファイルが生成される。一つは展示物である展示物オブジェクトの識別情報が記述されている XML ファイルである。もう一つは Unity がサポートしている言語である C# のファイルであり、これには博物館や展示物オブジェクトに関する位置情報などが記述されている。この二つのスクリプトファイルと展示物オブジェクトを指定の Unity プロジェクトにインポートし実行することにより、クエリで指定した 3次元仮想博物館が生成される。この過程の全体図を図 1 に示す。

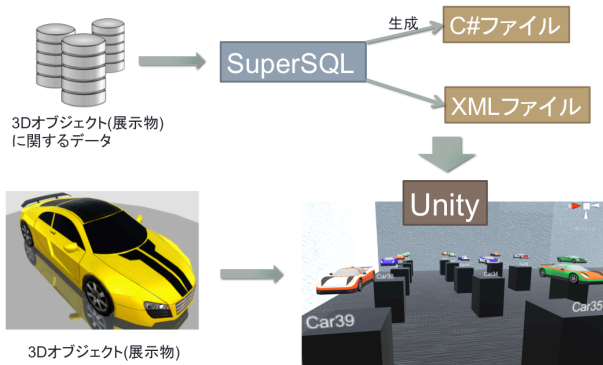


図 1 3次元仮想博物館生成の概要

4.2 質問文

3次元仮想博物館を生成する質問文は以下のようになる。

```

GENERATE Unity
  [分類項目,[id@{name='表示名の属性'}]
  (反復子 1)](反復子 2)
  @{ オブジェクト指定 }
  (結合子指定)
  ...
FROM 表
WHERE 条件
  
```

(2行目)

分類項目には属性名を書き、これによってその属性ごとに分類した部屋を生成する。idには、データベースで展示物オブジェクトのidを指定している属性を指定する。デフォルトでは、各展示物に表示する展示物の名前はidだが、装飾子を用いると、

表示名の属性を指定することによって表示する展示物名を変更できる。

(3行目)

反復子 1によって展示物オブジェクトの並べ方を指定し、反復子 2によって、部屋の並べ方を指定する。

(4行目)

装飾子によって博物館の建物や展示台のオブジェクトを変更する。指定がなかった場合はデフォルトで指定されているフォルダ内にあるオブジェクトが用いられる。

(5行目)

反復子 2でグルーピングすることによって生成した博物館が複数存在する場合、それらはこの結合子によって連結する。

4.3 3次元仮想空間の生成例

以下に3次元仮想空間の生成例として質問文とその結果を示す。用いるデータベースとして、company テーブル、furniture テーブル、cars テーブルを用いる。

- company テーブル (属性: id, name, year, emp_num)
name は会社の名前, year は設立年, emp_num は従業員数を表す。
- cars テーブル (属性: id, name, company, price, color, model)
name は車の名前, company は製造会社, price は値段, color は色, model は車種を表す。
- furniture テーブル (属性: id, name, category, company, color)
name は家具の名前, category は種類, company は製造会社, color は色を表す。

• 例 1

```

GENERATE Unity
  [category,[id@{name='name'}],!
  @{museum='Type2', stand='Type2'}
FROM furniture;
  
```

この質問文により、まずデータベースから furniture テーブルの情報が取得される。そして furniture テーブルの属性である category によって分類した部屋が生成され、それらは反復子 (!) によって y 軸方向に並べられる。部屋の中にある展示物オブジェクトは、反復子 (,) によって x 軸方向に並べられる。装飾子を用いることによって表示する展示物名を属性 name の値に変更し、博物館の建物と展示台はそれぞれ Type2 に変更されている。この生成結果を図 2 に示す。

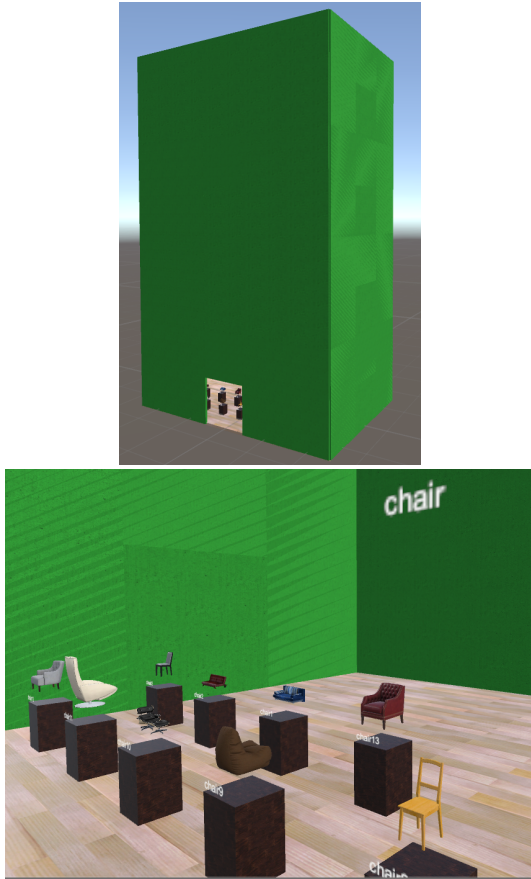


図 2 例 1 の生成結果

• 例 2

GENERATE Unity

```
[f.category,[f.id@{name='f.name'}],]! %
[c.model,[c.id@{name='c.name'}]]%,
```

FROM furniture f, cars c;

この質問文は反復子 (%) によって、二つの博物館が z 軸方向に連結されたものとなっている。まずデータベースから furniture テーブルと cars テーブルの情報が取得される。一つ目の博物館では、furniture テーブルの属性である category によって分類した部屋が生成され、それらは反復子 (!) によって y 軸方向に並べられる。部屋の中にある展示物オブジェクトは、反復子 (,) によって x 軸方向に並べられる。次に二つ目の博物館では、cars テーブルの属性である model によって分類した部屋が生成され、それらは反復子 (,) によって x 軸方向に並べられる。部屋の中にある展示物オブジェクトは、反復子 (%) によって z 軸方向に並べられる。この生成結果を図 3 に示す。

4.4 アーキテクチャ

SuperSQL には SSQL クエリを受け取り SQL クエリと表の構造 (TFEtree) に分けるパーザ、SQL クエリによる取得結果と表の構造情報を受け取りデータの整形をするデータコンストラクタ、整形されたデータを受け取り実際にファイルとして出力するコード生成部の三つの機構がある。図 4 にあるように本論文では、コード生成部を拡張することにより、SSQL クエリ

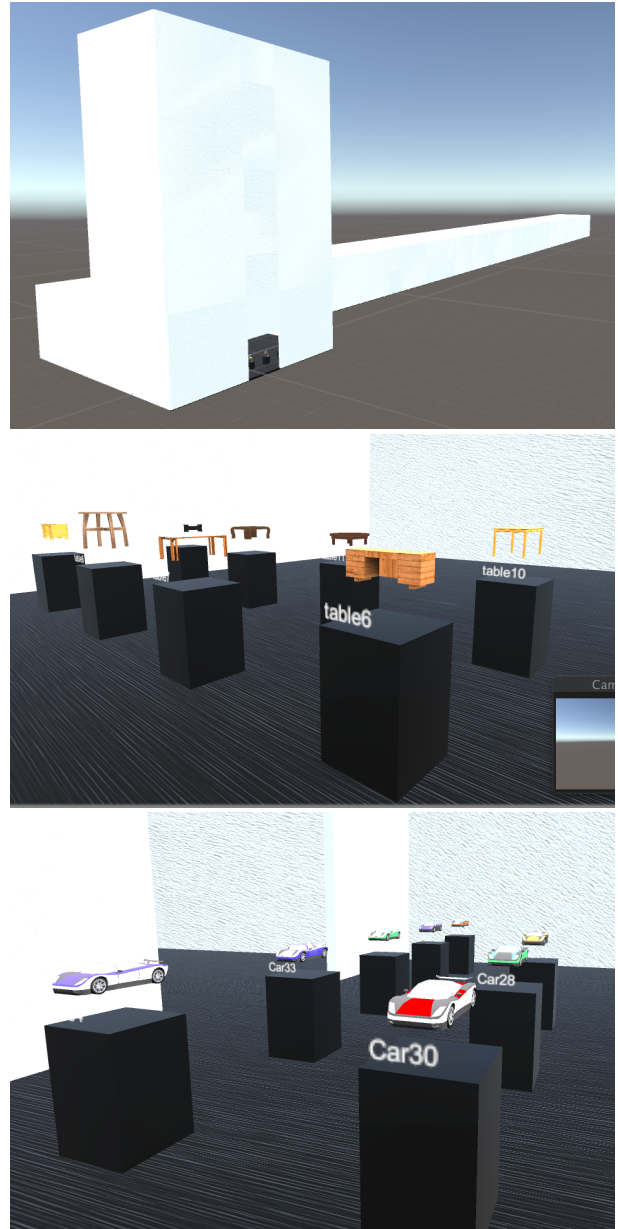


図 3 例 2 の生成結果

をパーザとコード生成部で解析すると C# と XML ファイルが作成されるアーキテクチャとなっている。

4.5 処理アルゴリズム

コード生成部の処理アルゴリズムについて説明していく。

4.5.1 XML 生成について

アルゴリズム 1 について説明していく。XML ファイル生成の際の入力はツリー構造のデータ、レイアウト式であり、出力は XML ファイルである。

コード生成部は入力を受け取ると、初めに XML タグを付加する。次に、まだ生成する博物館がある間は博物館タグを付加していく。同様に、まだ生成するフロアがある間はフロアタグを付加し、展示物 1 に連結して配置する他の展示物 2 が存在する間は展示物連結タグを付加し、生成する展示物がある間は展示物タグを付加していく。

4.5.2 C# 生成について

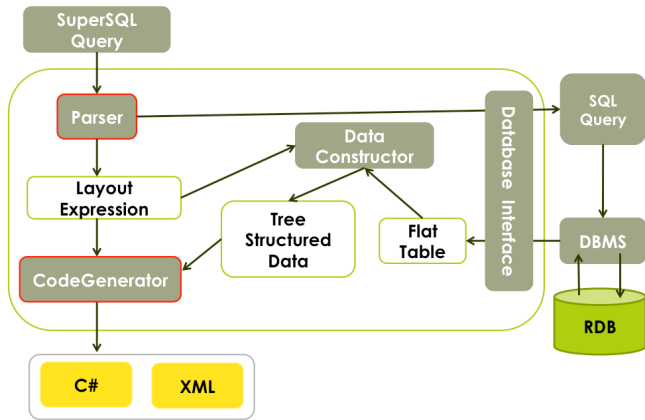


図 4 3次元仮想博物館生成のアーキテクチャ

Algorithm 1 XML ファイルの生成

Input: ツリー構造のデータ, レイアウト式

Output: XML ファイル

```

1: XML タグ付加
2: while 博物館の要素がある間 do
3:   博物館タグ付加
4:   while フロアの要素がある間 do
5:     フロアタグ付加
6:     while 展示物連結要素がある間 do
7:       展示物連結タグ付加
8:       while 展示物の要素がある間 do
9:         展示物タグ付加
10:      end while
11:    end while
12:  end while
13: end while
14: return XML ファイル

```

Algorithm 2 C#ファイルの生成

Input: レイアウト情報

Output: C#ファイル

```

1: while 博物館の要素がある間 do
2:   変数定義コード付加
3:   XML ファイル解析コード付加
4:   展示台, 展示物などの博物館内のオブジェクトを生成し, 位置
   などを調整するコードを付加
5:   Size 関数を用いてオブジェクトのサイズを調整するコードを付加
6:   博物館の建物生成コード付加
7: end while
8: Size 関数付加
9: return C#ファイル

```

アルゴリズム 2 について説明していく。C#ファイル作成の際の入力はレイアウト情報であり、出力は C#ファイルである。コード生成部は入力を受け取ると、生成する博物館がある間は、変数定義のコード、XML ファイル解析のコード、博物館内のオブジェクトを生成、位置調整などをするコード、オブジェクトサイズ調整コード、博物館の建物を生成するコードを付加する。Size 関数は、オブジェクトのサイズを測定し、全てのオブジェクトがほぼ同じ大きさになるよう調整する関数である。生

成する博物館がなくなると、Size 関数を付加する。

4.6 ヘッドマウントディスプレイの使用

本研究では、生成した 3次元仮想空間は HTC Vive というヘッドマウントディスプレイ (HMD) を用いて表示している。HMD は没入型であるので、平面上で表示する場合より 3次元仮想空間をよりリアルに体験することが可能であり、それによってユーザの理解度をさらに高めることができると考えられる。

5. 評価・検討

本論文で提案するシステムの有用性を評価するため、2つの方法で評価を行った。1つ目はデータベースのサイズと博物館の個数によって、本システムを用いて仮想空間を生成した場合と、用いなかった場合のコード量を比較した。2つ目は従来の仮想空間作成ツールとの機能の比較を行った。

5.1 コード量の比較

5.1.1 データベースのサイズでの比較

データベースのサイズ (展示物の数) を変更する場合において、部屋が 4 つありそれらを y 軸方向に並べた、4 階建ての博物館を SuperSQL の質問文を用いて作成した場合と、C#スクリプトを用いて作成した場合でコード量の比較を行った。表 1 に DB サイズが 10,100,300,500 件の場合の SuperSQL の質問文と C#スクリプトの行数を示す。

表 1 データベースのサイズを変更した際の行数の比較

DB サイズ	SuperSQL	C#
10 件	3	250
100 件	3	2050
300 件	3	6050
500 件	3	10050

表 1 より、データベースのサイズが大きくなるほど、SuperSQL 質問文を利用することが効果的であると分かる。

5.1.2 生成する博物館の個数での比較

データベースのサイズ (展示物の数) が 50 件である場合において、SuperSQL 質問文を用いた場合と C#スクリプトを用いた場合で特定の博物館を複数作成していき、それぞれの場合におけるコード量の比較を行った。表 2 に作成した博物館の個数が 1,5,10,15 個の場合の SuperSQL の質問文と C#スクリプトの行数を示す。

表 2 博物館の個数を変更した際の行数の比較

個数	SuperSQL	C#
1 個	3	1050
5 個	7	1226
10 個	12	1446
15 個	17	1666

表 2 より、生成する博物館の個数が多くなるほど、SuperSQL 質問文を利用することが効果的であると分かる。

よって、表1、表2より、本システムを利用するにはデータベースを準備するという初期コストがかかるが、複数の博物館を作成したい、展示するオブジェクトの数が多すぎる、様々なレイアウトの博物館を作成していきたいという場合には、本システムは効率的であるとわかる。

5.2 従来のツールとの機能比較

本論文で提案したシステムの仮想空間生成機能の性能を測るために、従来のツールとの比較を行った。今回比較に用いるツールは、本システムを構築するために用いたUnityと、ゲームエンジンを内蔵する3次元コンピュータグラフィックスソフトウェアのBlenderである。この2つを比較対象として選択した理由として、2つとも仮想空間を作成することに特化しているゲームエンジンであることと、現在多くのユーザに利用されているツールであることが挙げられる。

本論文で提案したシステム、Unity、Blenderの三段階での比較を表3に示す。

表3 各ツールの比較表

機能	本システム	Unity	Blender
データベース接続			x
アセットの形式			
プラットフォーム			
HMDへの接続			x
アニメーション	x		

表3より、本システムは、データベース接続、様々な形式のアセットへの対応、HMDへの接続の点で優れており、アニメーションの点で劣っている。それぞれの評価について詳しく述べていく。

データベース接続の点では本システムは優れている。Unityは他のツールと連携することによってデータベース接続が可能となっているのでとなっている。アセットに関しては、UnityもBlenderも様々な形式のアセットに対応しており、また、本システムはこの点においてはUnityの性能に依存しているので、Unityと同じ評価となっている。プラットフォームに関しては、UnityとBlenderは様々なプラットフォームに対応している。しかし本システムは、仮想空間を生成するまでのプラットフォームはUnityの性能に依存しているが、仮想空間を表示するために用いているHTC ViveというHMDは現状ではMicrosoft Windowsにしか対応していないので、評価がとなっている。HMDへの接続は、UnityはプラグインをインポートすればすぐにHMDでUnityで作成した仮想空間を表示することができるので、本システムとUnityの評価をとした。アニメーションに関しては、UnityとBlenderは作成できるが、本システムは作成できないのでxとした。

6. まとめ

6.1 結論

本研究では、データベースに格納された情報から、SuperSQLを用いて3次元仮想博物館を生成するシステムを提案した。このシステムにより、SuperSQLの質問文から異なるレイアウト

の様々な3次元仮想空間を生成することが可能となった。簡単な質問文を記述することで複雑な3次元仮想空間を生成でき、また質問文を少し変更すれば異なる3次元仮想空間を生成できるという点で有用であるといえる。

6.2 今後の課題

現状では、3次元仮想博物館生成におけるSuperSQLの質問文は、二次までのネストが可能となっており、これによって展示物と部屋に関する指定ができる。これを三次までのネストを可能にすることによって、いくつかの部屋のまとまりであるフロアを生成し、このフロアに関する並べ方などの指定を可能にしていきたいと考えている。また、展示台を置く幅など、装飾子による細かいデザインの指定の幅を増やしていきたいと考えている。他にも、現時点では生成物は博物館に限られているが、博物館に限定しない仮想空間を生成したり、既存の仮想空間の一部に、このシステムによって生成された仮想空間を埋め込んだり、アニメーションの作成を可能にするなどの課題が考えられる。

文献

- [1] SuperSQL: <http://SuperSQL.db.ics.keio.ac.jp>
- [2] M. Toyama: "SuperSQL: An Extended SQL for Database Publishing and Presentation", Proceedings of ACM SIGMOD 98 International Conference on Management of Data, pp. 584-586, 1998
- [3] Unity マニュアル: <https://docs.unity3d.com>
- [4] 中西 厚友, 遠山 元道. "SuperSQLによる仮想3次元空間の自動生成", 慶應義塾大学, 2008.
- [5] Chairi Kiourt, Anestis Koutsoudis, George Pavlidis. "DynaMus: A fully dynamic 3D virtual museum framework", Journal of Cultural Heritage, Volume 22, pp. 984991, 2016.
- [6] Daniel Sacher, Benjamin Weyers, Torsten W. Kuhlen, Wolfram Luther. "An Integrative Tool Chain for Collaborative Virtual Museums in Immersive Virtual Environments", Collaboration and Technology, Volume 9334 of the series Lecture Notes in Computer Science, pp. 86-94, 2015.
- [7] A. Martina, A. Bottino. "Using Virtual Environments as a Visual Interface for Accessing Cultural Database Contents", International Conference of Information Science and Computer Applications, pp. 1-6, 2012.
- [8] Ciro Donalek, S.G. Djorgovski, Alex Cioc, Anwell Wan, Jerry Zhang, Elizabeth Lawler, Stacy Yeh, Ashish Mahabal, Matthew Graham, Andrew Drake, Scott Davidoff, Jeffrey S. Norris, Giuseppe Longo "Immersive and collaborative data visualization using virtual reality platforms" 2014 IEEE International Conference on Big Data, pp. 27-30, 2014.