平成17年度
卒業研究報告書

Qtによる電磁場解析用のGUIの研究

秋田工業高等専門学校　電気工学科

研究者名　藤原　友希

指導教員名　山本　昌志

目 次

要旨

本研究では，GUIを用いて計算領域を作成し，その計算領域をメッシュで分割し，ディスプレイ上に表示するプログラムを作成した．ここでのGUIの作成にはQtを用いた．また，Qtのプログラムからメッシュで分割するMeshGeneratorのプログラムや，画像として表示するOpenGLのプログラムを実行するために，UNIXの命令であるforkやexecを，用いた．プログラムはC++のクラスを中心に作成したため，分かり易いソースコードとなっている．

第1章　緒言

ユーザの利便性を考え，現代のアプリケーションプログラムではGUI(Graphical User Interface)はとても重要である．また我々の研究室では，加速器開発用の電磁場解析ソフトウェアの開発を進めており，最終的には商品化を目指しているのでこのGUIは必須である．また，我々のGUIを検出する全体の制御プログラムは入出力操作だけではなく，様々な電磁場解析のプログラムを統括して，制御する役割も担っている．そのため，本研究ではこの開発を行い，プログラムを作成した．

図1.1に本研究室で作成しているソフトウェアで用いられているプログラムの全体構成を示す．GUIの役割はこの全体構成のうち，計算領域の作成と各電磁場解析の統括制御を行う．本研究ではGUIの役割のうちの計算領域の作成を行うプログラムを作成した．

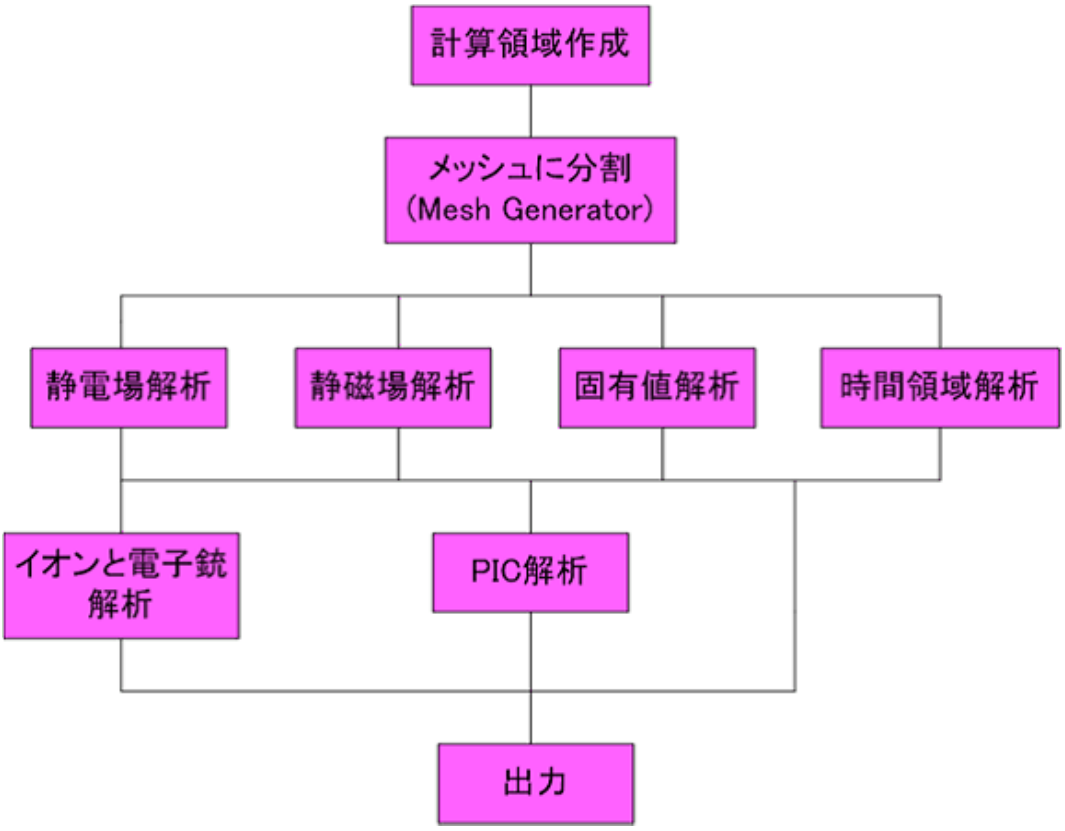

図 1.1: プログラムの全体構成

第2章　GUIツールキット Qt について

2.1　Qt とは

本研究では GUI のツールキットとして，Qt を用いた．以下にこの Qt について述べていく．Qt はノルウェーの企業 Trolltech 社が開発，配布を行っている C++クラスライブラリであり，Unix と X11 用の GUI ツールキットである [3]．そのため，Qt を用いて GUI を作成する際には特別な言語を習得する必要はなく，多数のプログラマが習得ずみの C++プログラミングを用いることで作成することが出来る．

また，Qt はその他の GUI ツールキットである Unix の Motif や，Windows の Windows-API よりも優れている [3]．その点について以下に述べていく．

Motif は Unix の世界での GUI ツールキットの中で最も有名であり，多くの主要な Unix ベンダが自社の OS(Operating System) 付属の GUI ツールキットとして採用してきた．この Motif は，単に GUI ツールキットというだけではなく，あるルックアンドフィールを規定する仕様でもあるため，Unix において間違いなく標準である．しかし，Motif はプログラムが極めて難解で，エラーを誘発しやすいという欠点がある．また Motif で作成したプログラムは，同じ動作をさせるために作成した Qt のプログラムよりも遙かに長くなる．

WindowsAPI には GUI 要素を生成したり，色を操作したりする関数が用意されている．WindowAPI を用いてプログラムを作成することは，いくらかは簡単であるが，それでも実用的なプログラムを作成するには何かと手間がかかる．MFC(Microsoft Foundation Classes) のような Windows 用の開発ツールは，高レベルな関数は用意されているが，洗練されたユーザインターフェンスを簡単に作成するために用いることは容易でない．そのためプログラマはユーザインターフェンスの作成に時間が膨大にかかり，アプリケーションの実質的な機能に集中する余裕がない．

以上の理由から Qt が両者より優れていることがわかり，Qt は前述のような欠点がないアプリケーションフレームワークの 1 つである．

フレームワークとは，ツールキットや GUI プログラミング API よりももう少し高度なものである．フレームワークは完全なプログラミングシステムで，エラーを誘発しやすい低レベルの細かい事柄をプログラマが気にしなくても済むようにしている．例を挙げると，キーボードのタイプや，マウスのクリックといったユーザの入力操作に対して，それを処理するために定義した関数や手続きなどを自動的に呼び出してくれるということである．

2.2 Qtの特徴

Qtには以下のような特徴がある [2] [4].

- 移植性が高い
- 高速である
- 非常に使いやすい
- 機能が豊富である
- C++のパフォーマンスを最大限に引き出すことが出来る
- フリーソフトもある

それぞれの特長について述べる．まず1つ目に「移植性が高い」ということは，Linuxだけではなく，Windows，Macintoshといった様々な環境でQtで作成したプログラムを実行することが出来る．そのため商品化を目指す際に，ターゲットとなる市場が広がるためとても重要である．

2つ目に「高速である」ということである．これはQtがそれぞれのプラットフォーム上のプリミティブな描画 API (Win 32 GDI、Xlib) で実装されているのため，高速な動作のアプリケーションを開発することが可能であるからである．これが他のアプリケーションであるJavaとの違いでもある．

3つ目に「非常に使いやすい」ということである．これは前述したように，他のGUIツールキットとは異なるアプリケーションフレームワークの1つだからであるため，ユーザの操作に対するコンピュータ側の処理を簡単に作成することができるからである．Qtではこのようなユーザの操作に対する処理のようなデータ通信に,「シグナル・スロット機構」を導入している．このことについては後述する．

4つ目に「機能が豊富である」ということである．Qtはマウスによるユーザの入力操作だけではなく，キーボードによる入力操作など様々な機能を持っている．以下にQtの機能の一部を紹介する．

- 描画する操作 (直線，矩形，ベジェ曲線など)
- キーボードによる操作 (タイプするキーの種類によって処理を変更)
- マウスによる操作 (左・右クリック，ダブルクリックなど)
- 音の操作 (タイマのアラームなど)
- ウインドウやメニューの操作 (ダイアログボックスやスクロールバーの作成など)
- 時間の操作 (タイマの時間の処理など)

などのような様々なものがある．

5つ目に「C++のパフォーマンスを最大限に引き出すことが出来る」ということである．これはQtがC++と非常に相性が良いことから明らかである．Qtはクラスライブラリなので，Qtのプログラム中ではクラスを非常に多く用いる．

6つ目に「フリーソフトもある」ということがある．フリーソフトであるため，Qtを誰でも簡単に手に入れることができ，プログラミングにコストがかからないという大きなメリットがある．

2.3 シグナル・スロット機構

Qt の最も重要な概念としてシグナル・スロット機構がある [3]．これらの関係について図 2.1 に示す．

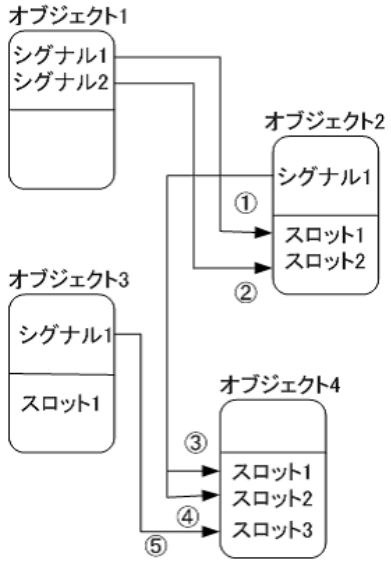

図 2.1: シグナルとスロットの関係

図 2.1 に示すようにシグナルとスロットは，オブジェクト間通信に用いられるものであり，この図では 5 つのシグナルとスロットの関係を示している．ここでオブジェクトとは，例としてマウスを挙げると，マウスの操作に対する関数や変数をひとまとめにしたものである．なぜここでオブジェクトを考えなければならないかというと，Qt は C++用の GUI ツールキットのため C++のオブジェクト指向 [5] をベースとしているためである．このオブジェクト指向の説明については割愛させてもらう．図 2.1 を眺めて分かるように，各オブジェクト間で接続されるシグナルとスロットは定義されている．このようにプログラマは，シグナル・スロット機構を使用する際には，プログラムコードでシグナルに対して特定のスロットを接続するように定義しなければならず，これを怠るとデータ通信を行うことが出来ない．また，1 つのシグナルに対しては複数のスロットを接続することが出来る．

以上がシグナル・スロット機構の概要である．それでは以下にシグナル，スロットとは何かについて述べていく．また説明をわかりやすくするために，シグナルとスロットの関係の具体的な例を図 2.2 に示す．

2.3.1 シグナル

シグナルとはオブジェクトの内部状態が変化した際に，発信される信号のことである．この信号が接続された特定のスロットに発信される．図 2.2 ではシグナルを発信するオブジェクトをマウスとし，マウスをクリックした時にシグナルが発信されることを示している．シグナルの例としては次のようなものがある．

- マウスやボタンのクリック (Qt の命令ではでは clicked())

- キーボードのタイプ (Qt では key())
- スライダーなどによる数値の変化 (Qt では valueChanged())

などがある．シグナルはユーザがオリジナルで作ることもできる．

2.3.2 スロット

スロットとは接続されたオブジェクトのシグナルが発信された際に，動作する関数のことである．図 2.2 ではスロットを持つオブジェクトをタイマとし，このオブジェクトのスロットはタイマのスタートやストップといった動作をさせる関数を指す．スロットも Qt 側であらかじめ用意されているものもあるが，ほとんどはユーザがオリジナルで作る．

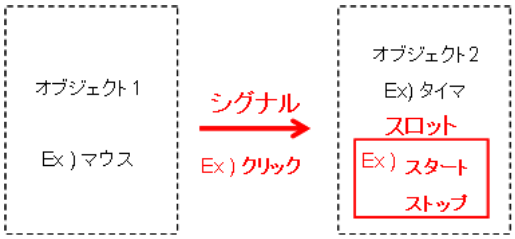

図 2.2: シグナルとスロットの具体例

Qt プログラムコードでこのシグナル・スロット機構を用いるには，以下のような命令を作成する．

$$QObject :: connect(OBJECT1, SIGNAL(signal1), OBJECT2, SLOT(slot1))$$

$OBJECT1$ はシグナルを送出するオブジェクト名，$signal1$ はスロットに接続するシグナル，$OBJECT2$ はシグナルを受け取るオブジェクト名，そして $signal2$ はシグナルに接続するスロットである．シグナルとスロットは，その名前とパラメータ型を記述し，それぞれを SIGNAL() と SLOT() で囲む．この機構を用いた例として，本研究のプログラム中の二点間を曲線で結ぶ際の円の半径を決定するボタンを挙げる．その時のウインドウを図 2.3 に示す．

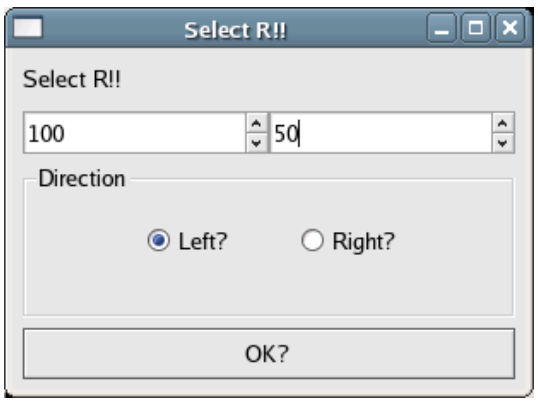

図 2.3: 曲線の半径を決めるウインドウ

その際の命令は次の通りである．

$$QObject :: connect(gobutton, SIGNAL(clicked()), this, SLOT(SlotGo()));$$

“*gobutton*” はプッシュボタンを示し，図 2.3 の”OK?” と書かれてあるボタンを示し，シグナルを発信するオブジェクトである．”this” は今いるオブジェクトを示している．この命令ではプッシュボタン”*gobutton*” をクリックすることで，今いるオブジェクト内で定義されている”*SlotGo*()” という半径を決定するスロットを呼び出されるという動作を示している．

第3章　1つのプログラムから別のプログラムを実行させる方法について

本研究ではQtプログラムで作成したウインドウ上で描いた計算領域を，MeshGeneratorというプログラム (有限要素法 [6] を用いている) でメッシュに分割し，メッシュに分割された計算領域を OpenGL のプログラムを用いて画像として表示している．そのため 1 つのプログラム (Qt のプログラム) から別のプログラム (MeshGenerator，OpenGL のプログラム) を実行させる必要がある．その方法について以下に示す．

3.1　別のプログラムを実行する手段

本研究ではその手段として UNIX の命令を用いた [1]．その手順を以下に示す．

1. 新しいプロセスを作成する (fork)
2. 作ったプロセス上で他のプログラムを実行させる (exec)
3. それらのプログラムが終了するのを待つ (wait)

ここでプロセスは，プログラムの実行単位である．つまり 1 つのプログラムを 1 つのプロセスと見なすことができ，本研究では Qt のプログラム，MeshGenerator のプログラム，OpenGL のプログラムがそれぞれ 1 つのプロセスである．

次に上記の操作をする UNIX の各命令について説明していく．UNIX の命令を用いた実際のプログラム Mesh.cpp，Meshout.cpp を付録 A に載せておく．図 3.1 に各 UNIX の命令と，その動作の関係を示す．

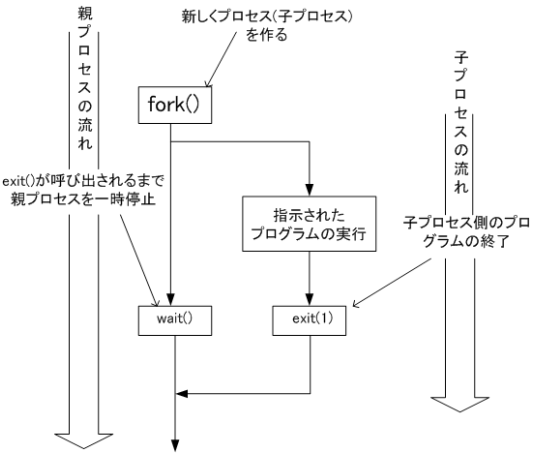

図 3.1: 別のプログラムを実行させる方法

3.1.1 プロセスの作成

新しいプロセスを作る命令はforkである．forkを実行することで，現在のプロセス(Qt)のコピーを，新しいプロセス(MeshGenerator or OpenGL)としてそれ自身のデータ空間を持つように作り，その2つのプロセスは並行して実行される．この命令を実行するプロセスは親プロセスと呼ばれ，新しく作られるプロセスは子プロセスと呼ばれる．これは図3.1を眺めても分かる．

forkの呼び出しに失敗した場合には-1が返され，成功した場合には親プロセスには子プロセスのプロセスIDを返し，子プロセスに対しては0を返す．

```
if((pid = fork()) < 0){
    perror("fork");
    exit(1);
   }

if(pid  == 0){
   execvp("../mesh2_0605/mesh2",argv);
   perror("../mesh2_0605/mesh2");
   exit(EXIT_SUCCESS);
   }
```

forkは本研究のプログラムでは上に示すように用いた．このプログラムでは戻り値が0より小さい(つまり-1)では，エラーであることを知らせ，子プロセスの呼び出しを終了する．しかし戻り値が0の場合には，子プロセスを呼び出し実行する事を示している．

3.1.2 プログラムの実行

プログラムを実行する命令は，一般にexecと総称される．このexecを呼び出すと，呼び出したプロセス上に新しいプログラムが読み込まれる．つまり，呼び出したプロセスが使っていたメモリは解放され，その場所に新しいプログラム(MeshGenerator or OpenGL)がロードされる．

execには様々な種類があるが，本研究ではexecvpを用いた．execvpでは，第一引数は実行されるプログラムへのパスを示す文字列で，第二引数は引数リストへのポインタを要素とするネルポインタで終わる配列である．execは本研究ではforkの場合と同様のプログラムで示すように用いた．この命令の第一引数は，一つ上の階層の$\mathrm{mesh2}_0605$というフォルダ内の$mesh2$という実行ファイルまでのパスを示している．次の行の$perror$や$exit$はこの$execvp$が失敗した際に呼び出される．

以上のfork，execを用いることで，新しく作ったプロセス上でMeshGeneratorまたはOpenGLのプログラムを読み込み，Qtのプログラムから MeshGenerator，OpenGLといった別のプログラムを実行することが出来るようになる．

3.1.3 プロセス終了までの待機

親プロセス，子プロセスは同時に実行されていき，お互いの動作には関与しない．このように，シェルはバックグラウンドでプロセスを開始する．しかしこの方法では親プロセ

スは，子プロセスの処理が終わるまで処理を続行できないことがよくある．そのためにwait という命令がある．wait が実行されると，プロセスがシグナルを受け取るか，または自分の子プロセスの 1 つが終了するまでそのプロセスは停止する．

もう 1 つ重要な命令は exit である．この命令は，exec の呼び出しが失敗した場合にのみ実行されるものである．もし exit がないと，新しく作られたプロセス上でプログラムが実行できなかった場合でも，常に親プロセスとは別に子プロセスが残ってしまい，子プロセスは親プロセス用に書かれた命令を実行し始め，正常に動作しなくなる．

本研究では exit は前ページのプログラムに示すように，プログラムが実行できなかった場合に用いられる．fork が失敗した場合には exit(1) とし，無条件でプログラムを終了させ，成功した場合は exit(EXIT_SUCCESS) として終了させる．wait は本研究のプログラム中では次のように用いられる．

```
while(wait(&status) == pid)
    ;
```

wait は，整数型変数のアドレスを引数とし，この引数には戻るときに子プロセスの終了ステータスが代入される．wait を実行すると，プロセスがシグナルを受け取るか，または自分の子プロセスの 1 つが終了するまでそのプロセスは停止する．上のプログラムでは，int 型の変数 pid に fork によって子プロセスのプロセス ID が代入されており，その値が変わらない (プロセスが終了しない) 限り wait の命令を実行し続けることを示している．そのため，親プロセスはその間停止し続けることになる．

第4章　Qtによるプログラム作成

4.1　プログラムの概要

本研究では，Qtを用いて電磁場解析を行う計算領域を作成し，MeshGenerator，OpenGLのプログラムを呼び出して，メッシュに分割し，分割された計算領域を表示するプログラムを作成した．使用した言語は，C++である．

上述したようにプログラムは，3つの構成からなっている．

1. マウスを用いてウインドウ上に計算領域を描き，データファイルを作る
2. メッシュ作成用のプログラムMeshGeneratorを呼び出し，メッシュデータファイルを作る
3. OpenGLを用いてメッシュで分割された計算領域を出力する

第5章　研究結果

Qt により作成したプログラムの計算領域入力ウインドウを図 5.1 に示す．マウス操作によって，図のような計算領域を指定することが出来る．

図に示すようにウインドウにはメニューバーがある．それぞれのメニューには次のような項目があり，各項目は特有の動作をする．

- File･･･Save(描かれた計算領域をデータファイルに変換)，Quit(プログラムの終了)
- Mesh･･･Cut in the Mesh(MeshGenerator を呼び出し，メッシュデータファイルを作る)，Draw Mesh(OpenGL を呼び出し，出力する)
- Select Line･･･Straight(直線で結ぶように設定)，Curve(曲線で結ぶように設定)
- SCALE･･･Select scale(計算領域のサイズの設定)

主に計算領域の設定には左クリックを用いる．ウインドウ上で左クリックを行うことで，シグナルが発信されクリックしたウインドウ上の座標値を保存するスロットを呼び出す．クリックする点が 2 点以上では新たなシグナルが発信され，メニューバーの Select Line の Straight を選択した場合は 2 点間を直線で結び，Curve を選択した場合は 2 点間を曲線で結ぶ．但し曲線で結ぶ場合は，その曲線の円の半径とその曲線の向き (左 or 右) を設定する．計算領域の作成において，最後の線だけはキーボードの「A」をタイプして描画する．こうすることで，全体の計算領域を再描画し，ある形状として認識できるようにする．

またこのプログラムではウインドウ上で作成した計算領域を編集することもでき，編集したい点の近傍で右クリックを行うことで編集用のボックスが表示され，そのボックス内に表示されている X,Y 座標値を変え，再びウインドウ上で「A」をタイプすることで編集が可能である．

後は，上述したメニューを用いて Save→Gut in the Mesh→Draw Mesh の順に選択することで，メッシュで分割された計算領域を出力する事が出来る．

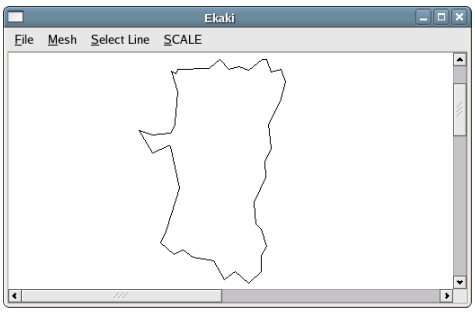

図 5.1: Qt によって作成されたウインドウ

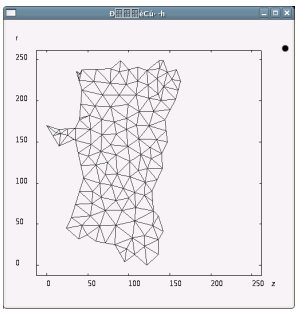

図 5.2: Qt で作成したモデルをメッシュできった図

図 5.2 はウインドウ上で描画した計算領域をメッシュに分割し，ディスプレイに表示している様子を示している．座標は円筒座標であり，計算領域は軸対象である．このようにメッシュで計算領域を分割することにより，各三角形の接点での電磁場の強さを計算し，その計算を計算領域全体の接点で行うことで計算領域全体での電磁場解析を行うことが出来る．

図 5.3 に本研究で作成した GUI のプログラムの全体構成を示す．この図で矢印はシグナルとスロットの関連を示している．

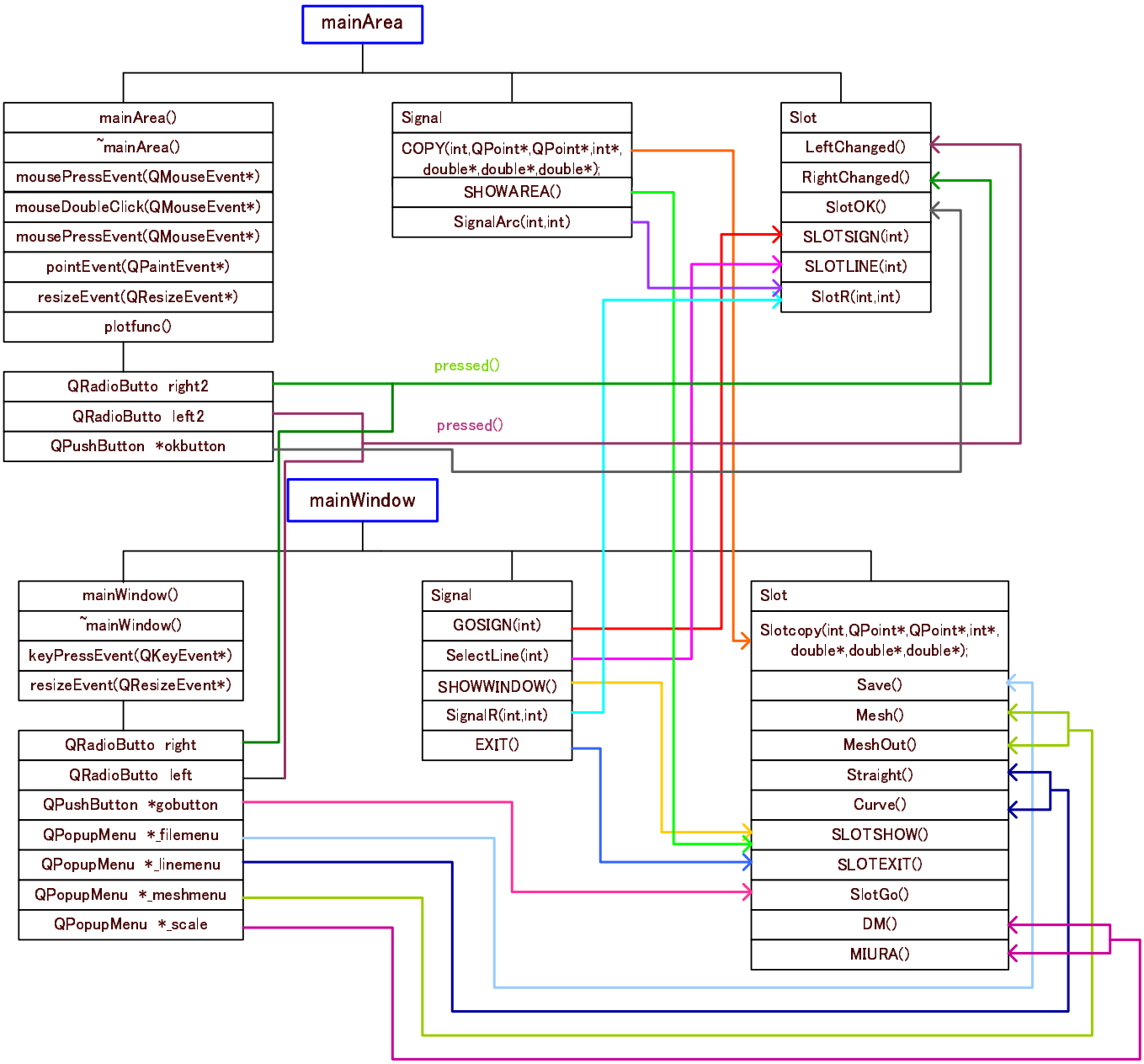

図 5.3: 別のプログラムを実行させる方法

第6章　結言

Qt を用いた GUI 環境で計算領域を描画し，その計算領域をメッシュで分割して，出力するプログラムの制作は成功した．

今後の課題として，GUI を用いて計算領域の作成は出来たが，全体のプログラムの統括制御をするまでには至らなかった．そのため，今後は電磁場解析用プログラムと連携するルーチンの開発を進めなくてはならない．

また，より複雑な形状の計算領域を作成するために，直線，曲線だけでなく描画ツールの種類を増やしていく必要がある．

本研究のプログラムの問題点を後輩のために残す．

1. 2 点間の x 座標が等しい時に，曲線を描こうとすると永久ループにはいる
2. 2 点間の x 座標がほぼ等しい時に，曲線を描くと曲線の向きが同じ方向になる
3. 時計回りに点をプロットしていかないと，メッシュデータファイルが作られない
4. 計算領域を作る線同士が交差した時に，メッシュデータファイルが作られない

これが主な欠点であると思われる．これ以外にもあるが，後は任せる．

関連図書

[1] David A.Curry. UNIX Cプログラミング. UNIX Systems Programming. ASCII, 1997.

[2] Jasmin Blanchette and Mark Summerfield. Qt GUIプログラミング. C MAGAZINE. SortBank Publishing, 2005.

[3] Mattbias Kalle Dalbeimer. Qtプログラミング入門. OREILLY, 1999.

[4] Daniel Solin. Qtプログラミング. 24時間集中講座. Pearson Education Japan, 2001.

[5] Tucker! オブジェクト指向開発講座. 憂鬱なプログラマのための. SHOEISHA, 2004.

[6] 戸川隼人. 変分法と有限要素法. 現代応用数学の基礎. 日本評論社, 1997.

謝辞

謝辞を述べる．まずこの秋田高専に入学させてくれた，中学校の担任，そして両親に感謝します．そして研究において，ご指導してくださった担当教員の山本昌志先生，さまざまな面でご助力していただいた専攻科の諸先輩方に対して感謝の気持ちで一杯です．

付 録A　GUI開発プログラム

リスト A.1: mainwindow.h

```
1  #ifndef MAINWINDOW_H
2  #define MAINWINDOW_H
3
4  #include<qapplication.h>
5  #include<qmenubar.h>
6  #include<qvbox.h>
7  #include<qspinbox.h>
8  #include<qmessagebox.h>
9  #include<qpopupmenu.h>
10 #include<qpainter.h>
11 #include<qscrollview.h>
12 #include<qwidget.h>
13 #include<qpixmap.h>
14 #include<stdlib.h>
15 #include<qevent.h>
16 #include<qlabel.h>
17 #include<qpushbutton.h>
18 #include<qlineedit.h>
19 #include<qscrollview.h>
20 #include<qstring.h>
21 #include<qradiobutton.h>
22
23
24
25 const int N = 100;//最大点数
26 const int P = 200;//最大曲線点数
27 const int shosu = 100;//少数値の最大値+1
28 const int MAX = 1000;//ウインドウ最大サイズ
29
30 //クラスの定義mainArea==========================================
31 class mainArea : public QWidget
32 {
33 Q_OBJECT
34
35    public:
36    mainArea();
37    ~mainArea();
38    void plotfunc( int c );
39
40   signals:
41    void COPY(int,QPoint*,QPoint*,int*,double*,double*,double*);
42    void SHOWAREA();
43    void SignalArc(int,int);
44
```

```
45    public slots:
46    void SLOTSIGN( int );
47    void SLOTLINE( int );
48    void SlotR( int,int );
49
50    private slots:
51    void LeftChanged();
52    void RightChanged();
53    void SlotOK();
54
55  protected:
56   virtual  void mousePressEvent(QMouseEvent *);
57   virtual  void mouseDoubleClickEvent(QMouseEvent*);
58   virtual  void paintEvent(QPaintEvent*);
59   virtual  void resizeEvent(QResizeEvent*);
60
61  private:
62    QPoint *point;
63    QPoint *cpoint;
64    QPoint *shosu_point;
65    QPixmap _buffer;
66    QColor _color;
67    QColor _currentcolor;
68    QSpinBox* x_spin;
69    QSpinBox* y_spin;
70    QSpinBox* x_shosu_spin;
71    QSpinBox* y_shosu_spin;
72    QSpinBox* R_spin;
73    QSpinBox* R_shosu_spin;
74    QHBox *hbox1;
75    QHBox *hbox2;
76    QHBox *hbox3;
77    QVBox *vbox1;
78    QRadioButton *left2;
79    QRadioButton *right2;
80    QButtonGroup *group;
81    QPushButton* okbutton;
82    QLabel* xlabel;
83    QLabel* ylabel;
84    QLabel* Rlabel;
85
86    int count;
87    int i,j,K,L;
88    bool down;
89    int sel[N+10];
90    int X,Y;
91    double R[N+10];
92    int r_seisu[N+10];
93    int r_shosu[N+10];
94    int ok;
95    char C;
96    int flag,sign,fugo;
97    int line;
98    int arc;
99    double cx,cy;
```

```
100     double Xc[N+10],Yc[N+10];
101     int plotX[N+10][P+10];
102     int plotY[N+10][P+10];
103   };
104   //=============================================================
105   //クラスの定義mainWindow=======================================
106   class mainWindow : public QWidget
107   {
108   Q_OBJECT
109
110   public:
111     mainWindow();
112    ~mainWindow();
113
114    public slots:
115    void Slotcopy(int,QPoint*,QPoint*,int*,double*,double*,double*);
116
117    signals:
118    void GOSIGN( int );
119    void SelectLine( int );
120    void SHOWWINDOW();
121    void SignalR( int,int  );
122    void EXIT();
123
124   protected:
125     virtual void resizeEvent(QResizeEvent*);
126     virtual void keyPressEvent(QKeyEvent*);
127
128     private slots:
129      void Save();
130      void Mesh();
131      void MeshOut();
132      void Straight();
133      void Curve();
134      void SLOTSHOW();
135      void SLOTEXIT();
136      void SlotGo();
137      void DM();
138      void MIURA();
139
140   private:
141     QPopupMenu* _filemenu;
142     QPopupMenu* _meshmenu;
143     QPopupMenu* _linemenu;
144     QPopupMenu* _scale;
145     QMenuBar* _menubar;
146     QPushButton* gobutton;
147     QPushButton* scok;
148     QScrollView* _scrollview;
149     QSpinBox* _scalevalue;
150     mainArea* _mainarea;
151     QPoint *point;
152     QPoint *shosu_point
153     QLabel* rlabel;
154     QVBox *vbox;
```

```
155     QVBox *vbox1;
156     QHBox *hbox;
157     QRadioButton *left;
158     QRadioButton *right;
159     QSpinBox *setR_spin;
160     QSpinBox *setR_shosu_spin;
161     QButtonGroup *group;
162 
163     int count;
164     int sign;
165     int sc;
166     int flag;
167     int line;
168     int sel[N+10];
169     double R[N+10];
170     double Xc[N+10],Yc[N+10];
171 };
172 //=============================================================
173 #endif //MAINWINDOW_H
```

リスト A.2: mainArea.cpp

```
1  #include<qapplication.h>
2  #include<qspinbox.h>
3  #include<qhbox.h>
4  #include<qvbox.h>
5  #include<qwidget.h>
6  #include<qlabel.h>
7  #include<qpushbutton.h>
8  #include<qbuttongroup.h>
9  #include<qradiobutton.h>
10 
11 #include "mainwindow.h"
12 
13 mainArea::mainArea()
14 {
15     //背景を白に設定=============================================
16     setBackgroundColor( white );
17     //=============================================================
18     //X座標編集用ボックスの設定=================================
19     vbox1 = new QVBox(0);
20     xlabel = new QLabel("X Cordinate (Integer Decimal)",vbox1);
21     //=============================================================
22     //Y座標編集用ボックスの設定=================================
23     hbox1 = new QHBox(vbox1);
24     ylabel = new QLabel("Y Cordinate (Integer Decimal)",vbox1);
25     hbox2 = new QHBox(vbox1);
26     //=============================================================
27     //X座標編集用スピンボックスの設定===========================
28     x_spin = new QSpinBox(hbox1);
29     x_shosu_spin = new QSpinBox(hbox1);
30     //=============================================================
31     //Y座標編集用スピンボックスの設定===========================
32     y_spin = new QSpinBox(hbox2);
33     y_shosu_spin = new QSpinBox(hbox2);
```

```
34    //=============================================================
35    //半径値編集用ボックスの設定=================================
36    Rlabel = new QLabel("R Cordinate (Integer Decimal)",vbox1);
37    hbox3 = new QHBox(vbox1);
38    //=============================================================
39  //半径値編集用スピンボックスの設定===========================
40    R_spin = new QSpinBox(hbox3);
41    R_shosu_spin = new QSpinBox(hbox3);
42
43    vbox1 -> setCaption("Cordinates Change!!");
44    vbox1 -> setMargin(6);
45    vbox1 -> setSpacing(6);
46    vbox1 -> setGeometry(600,50,300,200);
47    //=============================================================
48    //実行ボタンの設定 ===========================================
49    okbutton = new QPushButton("OK?",vbox1);
50    //=============================================================
51    //曲線方向の設定ラジオボタン()===============================
52    group = new QButtonGroup("Direction",vbox1);
53
54    left2 = new QRadioButton("Left?",group);
55    left2 -> move(65,30);
56    right2 = new QRadioButton("Right?",group);
57    right2 -> move(155,30);
58
59    group -> insert( left2 );
60    group -> insert( right2 );
61    //=============================================================
62    //各スピンボックスの初期値設定===============================
63    R_spin->setRange(0,MAX);
64    x_spin->setRange(0,MAX);
65    y_spin->setRange(0,MAX);
66    x_shosu_spin->setRange(0,shosu);
67    y_shosu_spin->setRange(0,shosu);
68    R_shosu_spin->setRange(0,shosu);
69    //=============================================================
70    //各変数の初期化=============================================
71    flag = 0;
72    line = 0;
73    count = 1;
74    sign = 0;
75    ok = 0;
76    K = 0;
77    L = 0;
78    cx = 0;
79    cy = 0;
80    down = TRUE;
81    point = new QPoint[N];
82    cpoint = new QPoint[P+10];
83    shosu_point = new QPoint[N];
84
85    for(int i=0;i>=N;i++){
86      shosu_point[i].setX(0);
87      shosu_point[i].setY(0);
88    }
```

```
89      //=============================================================
90      //シグナル・スロットの設定=====================================
91      QObject::connect(left2 ,SIGNAL(pressed()),this ,SLOT(LeftChanged()));
92      QObject::connect(right2 ,SIGNAL(pressed()),this ,SLOT(RightChanged()));
93
94      QObject::connect(okbutton,SIGNAL(clicked()),this ,SLOT(SlotOK()));
95      //=============================================================
96  }
```

リスト A.3: mainWindow.cpp

```
1   #include<qapplication.h>
2   #include<qpainter.h>
3   #include<qwidget.h>
4
5   #include "mainwindow.h"
6
7   void mainWindow::keyPressEvent(QKeyEvent* event)
8   {
9     QPainter paint;
10    QPainter buffer;
11     switch(event->key()){
12       //type no shurui wo kimeru=================================
13     case Key_A:
14       //sign no kettei===========================================
15       sign = 1;
16       emit GOSIGN( sign );
17       sign = 0;
18       //=========================================================
19       break;
20         }
21     //=============================================================
22  }
```

リスト A.4: Line.cpp

```
1   #include "mainwindow.h"
2   //chokusen ni settei=======================================
3   void mainWindow::Straight()
4   {
5     line = 1;
6     emit SelectLine( line );
7   }
8   //=============================================================
9   //curve ni settei==========================================
10  void mainWindow::Curve()
11  {
12    emit SHOWWINDOW();
13    line = 2;
14    emit SelectLine( line );
15  }
16  //=============================================================
17  //box wo hyozi=============================================
18  void mainWindow::SLOTSHOW()
19  {
```

```
20     vbox -> show();
21 }
22 //=============================================================
```

リスト A.5: Mesh.cpp

```
1  #include<stdio.h>
2  #include<unistd.h>
3  #include<sys/wait.h>
4  #include<sys/types.h>
5  #include<stdlib.h>
6  #include<string.h>
7  #include"mainwindow.h"
8
9  void mainWindow::Mesh()
10 {
11     int pid,status;
12     char *argv[10];
13
14     argv[1] = "ekaki";
15     argv[2] = NULL;
16
17     // Get a child process
18     if((pid = fork()) < 0){
19      perror("fork");
20      exit(1);
21     }
22
23     // The Child exectutes the code inside the if
24     if(pid  == 0){
25     execvp("../mesh2_0605/mesh2",argv);
26     perror("../mesh2_0605/mesh2");
27     exit(EXIT_SUCCESS);
28     }
29
30     // The parent exectuthe wait
31     while(wait(&status) == pid)
32       ;
33 }
```

リスト A.6: Meshout.cpp

```
1  #include<stdio.h>
2  #include<unistd.h>
3  #include<sys/wait.h>
4  #include<stdlib.h>
5  #include<string.h>
6  #include"mainwindow.h"
7
8  void mainWindow::MeshOut()
9  {
10     int pit,status;
11     char *argc[10];
12     argc[1] = "ekaki";
13     argc[2] = NULL;
```

```
14
15      //Get a child process
16      if((pit = fork()) < 0){
17        perror("fork");
18        exit(1);
19    }
20
21      if(pit == 0){
22        execvp("../DrawMesh/DrawMesh",argc);
23        perror("../DrawMesh/DrawMesh");
24        exit(1);
25    }
26
27    //The parent exectutes the wait
28    while(wait(&status) != pit)
29      ;
30  }
```

リスト A.7: PaintEvent.cpp

```
1  #include<qapplication.h>
2  #include<qpainter.h>
3  #include<qwidget.h>
4
5  #include "mainwindow.h"
6
7  void mainArea::paintEvent(QPaintEvent*)
8  {
9
10   QPainter paint;
11   QPainter buffer;
12
13   paint.begin(this);
14   buffer.begin(&_buffer);
15
16
17
18   for(int i=1;i<count-1;i++){
19     //ten wo chokusen de musubu=============================================
20     if(sel[i] == 1){
21   paint.drawLine(point[i],point[i+1]);
22   buffer.drawLine(point[i],point[i+1]);
23     }
24     //=======================================================================
25     //ten wo kyokusen de musubu============================================
26     if(sel[i] == 2){
27   for(int j=1;j<=P;j++){
28     paint.drawLine(plotX[i][j],plotY[i][j],plotX[i][j+1],plotY[i][j+1]);
29     buffer.drawLine(plotX[i][j],plotY[i][j],plotX[i][j+1],plotY[i][j+1]);
30
31   }
32     }
33     //=======================================================================
34   }
35   //saigo no line wo musubu==============================================
```

```
36      if(ok == 1){
37      paint.drawLine(point[count-1],point[1]);
38      buffer.drawLine(point[count-1],point[1]);
39      }
40      //=============================================================
41      paint.end();
42      buffer.end();
43      bitBlt(this,0,0,&_buffer);
44
45  }
```

リスト A.8: ResizeEvent.cpp

```
1   #include<qpixmap.h>
2
3   #include "mainwindow.h"
4   //window no size no henka ni awaseru=============================
5   void mainArea::resizeEvent(QResizeEvent* event)
6   {
7     QPixmap save(_buffer);
8     _buffer.resize(event->size());
9     _buffer.fill(white);
10    bitBlt(&_buffer,0,0,&save);
11  }
12  //=============================================================
```

リスト A.9: Save.cpp

```
1   #include<stdio.h>
2   #include<math.h>
3
4   #include "mainwindow.h"
5   //window zyo no model wo deta ni kaeru==========================
6   void mainWindow::Save()
7   {
8     FILE *fp;
9     fp = fopen("ekaki.model","w");
10
11    fprintf(fp,"#-----------------------------\n");
12
13    fprintf(fp,"$global\n");
14    fprintf(fp,"scale 0.001\n");
15    fprintf(fp,"mesh_size 20\n");
16    fprintf(fp,"$end\n");
17
18    fprintf(fp,"#-----------------------------\n");
19
20    fprintf(fp,"$regions\n");
21    fprintf(fp,"ext ");
22    for(int i=1;i<=count;i++){
23    fprintf(fp,"%d ",i);
24    }
25    fprintf(fp,"\n");
26    fprintf(fp,"$end\n");
27
```

```
28    fprintf(fp,"#------------------------------\n");
29
30    fprintf(fp,"$lines\n");
31    fprintf(fp,"#No\t point\t\t b.c type\t value\t style\t x\t y\t r\n");
32
33
34    for(int j=1;j<=count;j++){
35      if(j != count){
36
37        if(sel[j] == 1){
38        fprintf(fp,"%d\t %d\t %d\t dirichlet\t 0\t str\n",j,j,j+1);
39        }
40        else if(sel[j] == 2){
41
42          fprintf(fp,"%d\t %d\t %d\t dirichlet\t 0\t cir\t %f\t %f\t %f\n",j
43          ,j,j+1,sc*Xc[j],sc*(MAX-Yc[j]),fabs(sc*R[j]));
44
45
46        }
47
48      }
49      else{
50        fprintf(fp,"%d\t %d\t %d\t dirichlet\t 0\t str\n",j,j,1);
51      }
52    }
53
54
55    fprintf(fp,"$end\n");
56
57    fprintf(fp,"#--------------------------------\n");
58
59    fprintf(fp,"$points\n");
60    fprintf(fp,"#No\t px\t py\n");
61
62
63    for(int k=1;k<=count;k++){
64
65
66      fprintf(fp,"%d\t %f\t %f\n",k,(double)(sc*(point[k].x()+
67      (double)shosu_point[k].x()/shosu)),(double)(sc*(MAX - (point[k].y()+
68      (double)shosu_point[k].y()/shosu))));
69
70    }
71    fprintf(fp,"$end\n");
72
73    fprintf(fp,"#--------------------------------\n");
74    fprintf(fp,"$end_data\n");
75
76    fclose(fp);
77
78  }
```

リスト A.10: copy.cpp

```
1  #include<qapplication.h>
```

```
2  #include<qmenubar.h>
3  #include<qspinbox.h>
4  #include<qhbox.h>
5  #include<qvbox.h>
6  #include<qspinbox.h>
7
8  #include "mainwindow.h"
9  //iroiro na slot no teigi=============================================
10 // class kan no hensu no ukewatasi=====================================
11 void mainWindow::Slotcopy(int c,QPoint* qpoint,QPoint* spoint,int S[N]
12 ,double r[N],double X[N],double Y[N])
13 {
14   count = c;
15   point = qpoint;
16   shosu_point = spoint;
17   for(int i=1;i<=count;i++){
18     sel[i] = S[i];
19     R[i] = r[i];
20     Xc[i] = X[i];
21     Yc[i] = Y[i];
22
23   }
24 }
25 //=====================================================================
26 //saibyouga unnun=====================================================
27 void mainArea::SLOTSIGN(int s)
28 {
29   sign = s;
30   ok = 1;
31   if(sign == 1){
32     int k;
33     _buffer.fill(white);
34     bitBlt(this,0,0,&_buffer);
35     k=j;
36     if(flag == 1){
37       point[k].setX(x_spin->value());
38       point[k].setY(y_spin->value());
39
40       shosu_point[k].setX(x_shosu_spin->value());
41       shosu_point[k].setY(y_shosu_spin->value());
42     if(sel[k] == 2){
43       K = k;
44       plotfunc(k+1);
45     }
46
47     if(sel[k-1] == 2){
48       K = k-1;
49       plotfunc(k);
50     }
51     }
52     flag = 0;
53     update();
54   }
55   sign = 0;
56 }
```

```
57  //=============================================================
58  //chokusen ni settei=========================================
59  void mainArea::SLOTLINE(int l)
60  {
61    line = l;
62  }
63  //=============================================================
64  //kyokusen no muki left======================================
65  void mainArea::LeftChanged()
66  {
67    fugo = -1;
68  }
69  //=============================================================
70  //kyokusen no right==========================================
71  void mainArea::RightChanged()
72  {
73    fugo = 1;
74  }
75  //=============================================================
76  //kyokusen no hankei settei==================================
77  void mainArea::SlotR( int r,int s )
78  {
79    r_seisu[K] = r;
80    r_shosu[K] = s;
81    R[K] = fugo * (r + (double)s/shosu);
82  }
83  //=============================================================
84  //R settei button no settei==================================
85  void mainWindow::SlotGo()
86  {
87    emit SignalR(setR_spin -> value(),setR_shosu_spin -> value());
88    emit EXIT();
89  }
90  //=============================================================
91  //box wo hyouzisasenakusuru==================================
92  void mainWindow::SLOTEXIT()
93  {
94    vbox -> close();
95  }
96  //=============================================================
97  //hankei henshu suru box wo hyozi============================
98  void mainArea::SlotOK()
99  {
100   if(arc == 1){
101     emit SignalArc(R_spin->value(),R_shosu_spin->value());
102   }
103   vbox1 -> close();
104 }
105 //=============================================================
106 //scale wo settei suru box wo hyozi==========================
107 void mainWindow::DM()
108 {
109   if(flag == 0){
110   _scalevalue = new QSpinBox(vbox1);
111   _scalevalue -> setValue(1);
```

```
112     }
113     vbox1 -> show();
114     QObject::connect(scok,SIGNAL(clicked()),this,SLOT(MIURA()));
115     flag++;
116 }
117 //=============================================================
118 //scale value wo settei=======================================
119 void mainWindow::MIURA()
120 {
121     sc = _scalevalue->value();
122     vbox1 -> close();
123 }
124 //=============================================================
125 //=============================================================
```

リスト A.11: mouseDoubleClick.cpp

```
1  #include "mainwindow.h"
2
3  void mainArea::mouseDoubleClickEvent(QMouseEvent *event)
4  {
5      //window wo moto ni modosu=============================
6      if(event->button() == LeftButton){
7          down = TRUE;
8          count = 1;
9          ok = 0;
10         K = 0;
11         erase();
12         _buffer.fill( white );
13
14         bitBlt(this,0,0,&_buffer);
15     }
16     //=============================================================
17     if(event->button() == RightButton){
18
19     }
20 }
```

リスト A.12: mousePressEvent.cpp

```
1  #include<stdio.h>
2  #include<math.h>
3  #include<qapplication.h>
4  #include<qmenubar.h>
5  #include<qspinbox.h>
6  #include<qpainter.h>
7  #include<qwidget.h>
8
9  #include "mainwindow.h"
10
11 void mainArea::mousePressEvent(QMouseEvent *event)
12 {
13
14     i=1;
15     arc = 0;
```

```
16    QPoint search;
17    QPainter paint(this);
18
19    if( down ){
20     if(event->button() == LeftButton){
21      point[count] = event->pos();
22      paint.drawPoint(event->pos());
23      //chokusen wo hiku=============================================
24      if(line == 1){
25   if(count>1){
26        sel[K] = 1;
27        paint.drawLine(point[count-1],point[count]);
28      }
29      }
30      //=============================================================
31      //kyokusen wo hiku=============================================
32      if(line == 2){
33            if(count>1){
34              sel[K] = 2;
35               plotfunc(count);
36
37               for(i=1;i<=P;i++){
38                    paint.drawLine(cpoint[i],cpoint[i+1]);
39                  }
40            }
41      }
42
43      emit COPY(count,point,shosu_point,sel,R,Xc,Yc);
44
45      K++;
46      count++;
47     }
48     //=============================================================
49    }
50
51    //hikaku no zahyou wo hyozi=====================================
52     if(event->button() == RightButton){
53       down = FALSE;
54       search = event->pos();
55       while(1){
56         if(i >= count){
57           break;
58         }
59
60          if(search.x() <= point[i].x()+20.0 && search.x() >= point[i].x()
61          -20.0 && search.y() <= point[i].y()+20.0 && search.y() >=
62          point[i].y()-20.0)
63      {
64         group -> show();
65         hbox3 -> show();
66         flag = 1;
67
68         j=i;
69         if(sel[i] == 2 || sel[i-1] == 2){
70           arc = 1;
```

```
71
72            R_spin->setValue(r_seisu[i]);
73            R_shosu_spin->setValue(r_shosu[i]);
74          }
75
76          x_spin->setValue(point[i].x());
77          y_spin->setValue(point[i].y());
78          x_shosu_spin->setValue(shosu_point[i].x());
79          y_shosu_spin->setValue(shosu_point[i].y());
80
81          if(arc == 0){
82            group -> hide();
83            hbox3 -> hide();
84          }
85          vbox1 -> show();
86
87          break;
88        }
89            i++;
90         }
91      }
92      //=================================================================
93  }
```

リスト A.13: plotfunc.cpp

```
1  #include<stdio.h>
2  #include<math.h>
3  #include<qlineedit.h>
4  #include<qvbox.h>
5  #include<qstring.h>
6  #include<qwidget.h>
7  #include "mainwindow.h"
8
9  void mainArea::plotfunc(int c)
10 {
11    double cx,cy;
12   double dx,dy;
13   double ddx,ddy;
14   double yy,xx,r;
15   int egg;
16   double  x,y;
17   double delta,delta21,delta22,delta31,delta32;
18   int k;
19   double katamuki,seppen;
20   double Xcor,Ycor;
21   double distance;
22
23   xx = 0.0;
24    r = 0.0;
25   dy = 0.0;
26   egg = 0;
27   katamuki = 0.0;
28
29
```

```
30          cx = (point[c-1].x() + point[c].x())/2.0;
31          cy = (point[c-1].y() + point[c].y())/2.0;
32          dx = cx;
33
34          ddx = fabs(point[c].x() -cx);
35          ddy = fabs(point[c].y() -cy);
36
37          yy = sqrt(ddx*ddx + ddy*ddy);
38
39          if(point[c-1].y() == point[c].y() || point[c-1].x() == point[c].x()){
40
41            katamuki = 0.0;
42          }
43          else{
44          katamuki =(double)(point[c].y() - point[c-1].y())/(point[c].x()
45          - point[c-1].x());
46          }
47          distance = (double)sqrt(pow(point[c].x() - point[c-1].x(),2)
48          +pow(point[c].y() - point[c-1].y(),2));
49
50           seppen = cy +cx/katamuki;
51
52           if(down == TRUE)
53             {
54             emit SHOWAREA();
55           }
56
57           while(1){
58             if(distance <= 2.0*fabs(R[K]))
59               {
60                 break;
61               }
62             else{
63             loop:
64               printf("This value isunuitable R's value!!\n");
65               printf("R=");
66               scanf("%lf",&R[K]);
67             }
68           }
69
70          if(katamuki != 0.0){
71          while(1){
72
73            if((fabs(fabs(R[K])-r))<0.01){
74              break;
75            }
76            if(R[K]<0){
77              if(fabs(R[K])>r){
78
79            dx += 0.01;
80            }
81              if(r>fabs(R[K])){
82              dx -= 0.001;
83
84              }
```

```
85              }
86              if(R[K]>0){
87                if(fabs(R[K])>r){
88                dx -= 0.01;
89
90                }
91                if(r>fabs(R[K])){
92                dx += 0.001;
93
94                }
95              }
96
97              dy = -dx/katamuki + seppen;
98              xx = sqrt((dx-cx)*(dx-cx) + (dy-cy)*(dy-cy));
99
100           r = sqrt( xx*xx + yy*yy );
101
102           }
103           }
104
105           if(katamuki == 0){
106
107             while(1){
108               if(fabs(fabs(R[K])-r)<0.01){
109                 break;
110               }
111               if(point[c-1].y() == point[c].y()){
112
113                 if(R[K]<0){
114
115                   if(fabs(R[K])>r){
116                 dy += 0.01;
117               }
118                   if(r>fabs(R[K])){
119                 dy -= 0.001;
120               }
121                 }
122                 //down totu
123                 if(R[K]>0){
124
125                   if(fabs(R[K])>r){
126                 dy -= 0.01;
127                 }
128                   if(r>fabs(R[K])){
129                 dy += 0.001;
130                 }
131                 }
132
133               r = sqrt( pow((point[c].x() - cx),2) + pow((dy-cy),2));
134               }
135               if(point[c-1].x() == point[c].x()){
136                 //left totu
137                 if(R[K]<0){
138                   if(fabs(R[K])>r){
139                   dx += 0.01;
```

```
140                 }
141                   if(r>fabs(R[K])){
142                   dx -= 0.001;
143                 }
144                 }
145                 //right totu[
146                 if(R[K]>0){
147                   if(fabs(R[K])>r){
148                 dx -= 0.01;
149                 }
150                   if(r>fabs(R[K])){
151                 dx += 0.001;
152                 }
153                 }
154                 r = sqrt(pow((dx-cx),2) + pow((point[c].y()  - cy),2));
155               }
156
157             }
158
159           }
160
161           Xc[K] = dx;
162           Yc[K] = dy;
163
164
165           //above chusin gime
166           x = point[c-1].x();
167           delta =(fabs(point[c-1].x()-point[c].x()))/P;
168
169           delta21 = 2.0*(fabs((Xc[K]-fabs(R[K]))-point[c-1].x()))/P;
170           delta22 = 2.0*(fabs((Xc[K]-fabs(R[K]))-point[c].x()))/P;
171
172           delta31 = 2.0*(fabs((Xc[K]+fabs(R[K]))-point[c-1].x()))/P;
173           delta32 = 2.0*(fabs((Xc[K]+fabs(R[K]))-point[c].x()))/P;
174
175           y = point[c-1].y();
176           k=1;
177           double a = 0;
178           if(point[c-1].y()<=Yc[K] && point[c].y()<=Yc[K] || point[c-1].y()
179          >=Yc[K] && point[c].y()>=Yc[K]){
180           while(1){
181             if( k>P+1 ){
182               break;
183             }
184
185             if(point[c-1].y()<=Yc[K] && point[c].y()<=Yc[K])
186               {
187                 if(point[c-1].x()<point[c].x()){
188
189                   a=sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
190                  y = Yc[K] - a;
191                  x += delta;
192                 }
193                 if(point[c-1].x()>point[c].x()){
194                   a=sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
```

```
195                 y = Yc[K] - a;
196                 x -= delta;
197               }
198                 if(y < 0){
199                   goto loop;
200                 }
201               }
202             else if(point[c-1].y()>=Yc[K] && point[c].y()>=Yc[K])
203               {
204                 if(point[c-1].x()<point[c].x()){
205                   a=sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
206                 y = Yc[K] + a;
207                 x += delta;
208                 }
209                 if(point[c-1].x()>point[c].x()){
210                   a=sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
211                 y = Yc[K] + a;
212                 x -= delta;
213                 }
214               }
215
216             Xcor = x;
217             Ycor = y;
218
219             //shisha gonyu
220             Xcor += 0.5;
221             Ycor += 0.5;
222
223             cpoint[k].setX((int)Xcor);
224             cpoint[k].setY((int)Ycor);
225
226             plotX[K][k] = cpoint[k].x();
227             plotY[K][k] = cpoint[k].y();
228             k++;
229           }
230           }
231
232           int A,B;
233           A = point[c-1].y();
234           B = point[c].y();
235           x = point[c-1].x();
236
237           if(A>Yc[K] && B<Yc[K] || A<Yc[K]&&B>Yc[K]){
238             while(1){
239
240               if(k>=P/2){
241                 break;
242               }
243               if(point[c-1].y()>Yc[K]){
244
245                 if(R[K] < 0){
246
247                 a = sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
248                 y = Yc[K] + a;
249                 x -= delta21;
```

```
250                 }
251                 if(R[K] > 0){
252                     a = sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
253                 y = Yc[K] + a;
254                 x += delta31;
255                 }
256               }
257               if(point[c-1].y()<Yc[K]){
258                  if(R[K] > 0){
259                  a = sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
260                 y = Yc[K] - a;
261                 x += delta31;
262                  }
263                if(R[K] < 0){
264                  a = sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
265                 y = Yc[K] - a;
266                 x -= delta21;
267                }
268               }
269             Xcor = x;
270             Ycor = y;
271
272             //shisha gonyu
273             Xcor += 0.5;
274             Ycor += 0.5;
275
276             cpoint[k].setX((int)Xcor);
277             cpoint[k].setY((int)Ycor);
278
279             plotX[K][k] = cpoint[k].x();
280             plotY[K][k] = cpoint[k].y();
281
282             k++;
283             }
284
285             if(k>=P/2){
286
287               while(1){
288                 if(k>P+1){
289                   break;
290                 }
291
292                   if(point[c-1].y()>Yc[K]){
293
294                     if(R[K] < 0){
295                   a = sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
296                   y = Yc[K] - a;
297                   x += delta22;
298                     }
299                  if(R[K] > 0){
300                   a = sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
301                   y = Yc[K] - a;
302                   x -= delta32;
303                 }
304                   }
```

```
305
306                     if(point[c-1].y()<Yc[K]){
307                     if(R[K] > 0){
308                     a = sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
309                    y = Yc[K] + a;
310                    x -= delta32;
311                     }
312                   if(R[K] < 0){
313                     a = sqrt(pow(fabs(R[K]),2)-pow(x-Xc[K],2));
314                    y = Yc[K] + a;
315                    x += delta22;
316                   }
317                     }
318                    Xcor = x;
319               Ycor = y;
320
321               //shisha gonyu
322               Xcor += 0.5;
323               Ycor += 0.5;
324
325               cpoint[k].setX((int)Xcor);
326               cpoint[k].setY((int)Ycor);
327
328               plotX[K][k] = cpoint[k].x();
329               plotY[K][k] = cpoint[k].y();
330
331               k++;
332                  }
333               }
334
335                   if(x < 0){
336                     goto loop;
337                   }
338            }
339            egg = 0;
340  }
```

リスト A.14: resizeEvent.cpp

```
1  #include<qpopupmenu.h>
2  #include<qpainter.h>
3  #include<qwidget.h>
4  #include<qpixmap.h>
5  #include<stdlib.h>
6  #include<qevent.h>
7  #include<qlabel.h>
8  #include<qpushbutton.h>
9  #include<qlineedit.h>
10 #include<qscrollview.h>
11 #include "mainwindow.h"
12
13 void mainWindow::resizeEvent(QResizeEvent* event)
14 {
15   _scrollview -> setGeometry(0,_menubar -> height(),width(),height()
16   - _menubar -> height());
```

```
17 }
```

リスト A.15: Mesh.cpp

```
1  #include<stdio.h>
2  #include<unistd.h>
3  #include<sys/wait.h>
4  #include<sys/types.h>
5  #include<stdlib.h>
6  #include<string.h>
7  #include"mainwindow.h"
8
9  void mainWindow::Mesh()
10 {
11     int pid,status;
12     char *argv[10];
13
14     argv[1] = "ekaki";
15     argv[2] = NULL;
16
17     // Get a child process
18     if((pid = fork()) < 0){
19      perror("fork");
20      exit(1);
21     }
22
23     // The Child exectutes the code inside the if
24     if(pid  == 0){
25     execvp("../mesh2_0605/mesh2",argv);
26     perror("../mesh2_0605/mesh2");
27     exit(EXIT_SUCCESS);
28     }
29
30     // The parent exectuthe wait
31     while(wait(&status) == pid)
32       ;
33 }
```

リスト A.16: Meshout.cpp

```
1  #include<stdio.h>
2  #include<unistd.h>
3  #include<sys/wait.h>
4  #include<stdlib.h>
5  #include<string.h>
6  #include"mainwindow.h"
7
8  void mainWindow::MeshOut()
9  {
10     int pit,status;
11     char *argc[10];
12     argc[1] = "ekaki";
13     argc[2] = NULL;
14
15     //Get a child process
```

```
16      if((pit = fork()) < 0){
17        perror("fork");
18        exit(1);
19    }
20
21      if(pit == 0){
22        execvp("../DrawMesh/DrawMesh",argc);
23        perror("../DrawMesh/DrawMesh");
24        exit(1);
25    }
26
27    //The parent exectutes the wait
28    while(wait(&status) != pit)
29      ;
30  }
```

リスト A.17: Qmain.cpp

```
1  #include<qapplication.h>
2  #include<qwidget.h>
3
4  #include "mainwindow.h"
5
6  int main(int argc,char* argv[])
7  {
8    QApplication myapp(argc,argv);
9    mainWindow* mywidget = new mainWindow();
10   myapp.setMainWidget(mywidget);
11   mywidget->show();
12   return myapp.exec();
13 }
```