void CALLBACK CBrainStimSerialArmDlg::Mtimer(UINT wTimerID, UINT msg,DWORD dwUser, DWORD dw1, DWORD dw2)

{

CBrainStimSerialArmDlg \*pTimer = (CBrainStimSerialArmDlg\*) dwUser;

/\* Impedance control

MaxonLib \*m = new MaxonLib;

short qddot4,qddot5; //input

int k = 30 ; float kd = 0.02; float ki = 0; // gain

int qd4,qd5; //desired

int q4, qdot4; //state

int q5, qdot5;

int e4, e5;

int intgr\_e4,intgr\_e5;

float diff\_q4, diff\_q5;

float sTime = 0.001;

qd4 = 10; qd5 = 10;

q4 = pTimer->m\_encoder4; q5 = pTimer->m\_encoder5;

e4 = ( qd4 - q4 ); e5 = ( qd5 - q5 );

intgr\_e4 = pTimer->intgrErr[3]; intgr\_e5 = pTimer->intgrErr[4];

diff\_q4= (float)((q4 - pTimer->befq[3]) / sTime);

diff\_q5= (float)((q5 - pTimer->befq[4]) / sTime);

qddot4 = k \* e4 + ki \* intgr\_e4 - kd \* diff\_q4;

qddot5 = -( k \* e5 + ki \* intgr\_e5 - kd \* diff\_q5 );

m->torqControl(MaxonHandle,4,qddot4);

m->torqControl(MaxonHandle,5,qddot5);

if(data\_count % 100 == 0)

{

cout << " Kp : " << k << " Kd : " << kd << " Ki : " << ki << endl;

cout << " qddot4 : " << qddot4 << endl;

cout << " qddot5 : " << qddot5 << endl;

}

pTimer->intgrErr[3] = e4 + pTimer->intgrErr[3];

pTimer->intgrErr[4] = e5 + pTimer->intgrErr[4];

pTimer->befq[3] = q4;

pTimer->befq[4] = q5;

int datasize = sizeof(data)/sizeof(int);

if(data\_count < datasize)

{

data[data\_count] = q4;

data2[data\_count] = q5;

}

else if(data\_count > datasize-1) // 데이터가 꽉 차면 flag 0으로 변경

{

file\_flag = 0;

}

data\_count++;

delete m;

\*/

MaxonLib \*m = new MaxonLib;

Kinematics \*k = new Kinematics;

///////////////// Demo ///////////////////

/\*struct PoseOfBody PoseD;

struct IKJoint InvK;

PoseD.x=pTimer->preDefine\_x[timerCount]; PoseD.y=pTimer->preDefine\_y[timerCount]; PoseD.z=pTimer->preDefine\_z[timerCount];

PoseD.rz=pTimer->preDefine\_rz[timerCount]; PoseD.ry=pTimer->preDefine\_ry[timerCount]; PoseD.rx=pTimer->preDefine\_rx[timerCount];

timerCount ++;

k->ik(PoseD.x,PoseD.y,PoseD.z,PoseD.rz,PoseD.ry,PoseD.rx,&InvK);

if(bIKFlag)

{

m->Move(MaxonHandle,3,75,800,1); // absolute

while(VCS\_GetMovementState(MaxonHandle, 3 , &(m->bTargetReached), &(m->m\_ulErrorCode)))

{

if(m->bTargetReached == 1)

{

cout<<"Next target. "<< endl;

break;

}

}

for(int i = 1; i<7; i++)

{

if( i == 3)

{

continue;

}

m->Move(MaxonHandle,i,InvK.InvJoint[i-1],1); // absolute

}

long ikjoint3 = InvK.InvJoint[2];

AfxBeginThread(movingCheck,(void\*)ikjoint3);

//while(m->MovingCheck(MaxonHandle))

//{

// if(m->bMovingCheck == 1)

// {

// cout<<"Start to move a joint 3."<<endl;

// break;

// }

//}

//m->Move(MaxonHandle, 3, InvK.InvJoint[2], 400, 1);

}

if(timerCount == 6)

{

cout << "Finished." <<endl;

pTimer->bTimer = timeKillEvent(pTimer->m\_idEvent);

}\*/

///////////////// Demo ///////////////////

Matrix4f T\_RBtoD,T\_CtoDesired;

k->GetTransform(Target\_q0,Target\_qx,Target\_qy,Target\_qz,Target\_tx,Target\_ty,Target\_tz,&T\_CtoDesired);

// Record 시 사용

/\*if( pTimer->bRecordFlag )

{

if(recordCount < 10000)

{

data[timerCount] = Target\_tx;

data2[timerCount] = Target\_ty;

data3[timerCount] = Target\_tz;

}

}

\*/

Matrix4f ROT\_X\_\_90;

k->Rotation(0,0,-90,&ROT\_X\_\_90);

T\_RBtoD = pTimer->T\_RBtoC \* T\_CtoDesired \* ROT\_X\_\_90;

/// --------------- 오차 측정 --------------- ///

Matrix4f T\_CtoEEMarker, T\_CtoEE;

k->GetTransform(q0[2],qx[2],qy[2],qz[2],tx[2],ty[2],tz[2],&T\_CtoEEMarker);

Matrix4f T\_EEMarkerToEE\_r,T\_EEMarkerToEE\_t,T\_EEMarkerToEE; // r: 회전변환, t: 위치변환

/// TMS ///////

pTimer->mx = 54; pTimer->my = -67.175; pTimer->mz = -67.175;

pTimer->mrz = 0; pTimer->mry = 0; pTimer->mrx = 135;

k->Translation(pTimer->mx,pTimer->my,pTimer->mz,&T\_EEMarkerToEE\_t); k->Rotation(pTimer->mrz,pTimer->mry,pTimer->mrx,&T\_EEMarkerToEE\_r);

// Ultrasound //

//pTimer->mx = pTimer->ml\*cos(pTimer->mTheta); pTimer->my = 0; pTimer->mz = -pTimer->ml\*sin(pTimer->mTheta);

//pTimer->mrz = 0; pTimer->mry = -45; pTimer->mrx = -90;

// Mini-TMS //

//k->Translation(58,0,-95,&T\_EEMarkerToEE\_t); T\_EEMarkerToEE = T\_EEMarkerToEE\_t;

k->Translation(pTimer->mx,pTimer->my,pTimer->mz,&T\_EEMarkerToEE\_t); k->Rotation(pTimer->mrz,pTimer->mry,pTimer->mrx,&T\_EEMarkerToEE\_r);

T\_EEMarkerToEE = T\_EEMarkerToEE\_t \* T\_EEMarkerToEE\_r; T\_CtoEE = T\_CtoEEMarker \* T\_EEMarkerToEE;

Matrix4f T\_RBtoEE;

T\_RBtoEE = pTimer->T\_RBtoC \* T\_CtoEE; // Pose of end-effector w.r.t robot base

/// Position error

Vector3d err\_p(T\_RBtoD.coeff(0,3) - T\_RBtoEE.coeff(0,3),T\_RBtoD.coeff(1,3) - T\_RBtoEE.coeff(1,3),T\_RBtoD.coeff(2,3) - T\_RBtoEE.coeff(2,3));

/// Orientation error

Matrix3f R\_RBtoEE,R\_err;

R\_RBtoEE = T\_RBtoEE.block<3,3>(0,0).transpose();

R\_err = R\_RBtoEE\*T\_RBtoD.block<3,3>(0,0);

Orien ori;

k->GetEuler(&R\_err,&ori);

Vector3d err\_o(ori.rz,ori.ry,ori.rx);

/// Derivative of end-effector

double diff\_px,diff\_py,diff\_pz;

diff\_px = (T\_RBtoEE.coeff(0,3) - pTimer->bef\_px) / pTimer->samplingTime;

diff\_py = (T\_RBtoEE.coeff(1,3) - pTimer->bef\_py) / pTimer->samplingTime;

diff\_pz = (T\_RBtoEE.coeff(2,3) - pTimer->bef\_pz) / pTimer->samplingTime;

if((bFlag[2])&&(bFlag[3])) // 마커가 둘다 보일 때만 피드백하기.

{

T\_RBtoD.coeffRef(0,3) = Kp\*T\_RBtoD.coeff(0,3) + Ki\*err\_p.coeff(0) - Kd\*diff\_px;

T\_RBtoD.coeffRef(1,3) = Kp\*T\_RBtoD.coeff(1,3) + Ki\*err\_p.coeff(1) - Kd\*diff\_py;

T\_RBtoD.coeffRef(2,3) = Kp\*T\_RBtoD.coeff(2,3) + Ki\_Z\_axis\*err\_p.coeff(2) - Kd\*diff\_pz;

}

pTimer->bef\_px = T\_RBtoEE.coeff(0,3); pTimer->bef\_py = T\_RBtoEE.coeff(1,3); pTimer->bef\_pz = T\_RBtoEE.coeff(2,3);

/// ^^^^^^^^^^^ 오차 측정 ^^^^^^^^^^^ //

/// 구면좌표계 실험 시 사용

/\*

Error in Spherical coordinate

Vector3d V\_RBtoDW(Target\_tx-104\*T\_RBtoD.coeff(0,3),Target\_ty-104\*T\_RBtoD.coeff(1,3),Target\_tz-104\*T\_RBtoD.coeff(2,3));

Vector3d

SpheCoordi des\_sphe,esti\_sphe;

k->CarteToSphe(Target\_tx-l6\*T\_RBtoD.coeff(0,0),Target\_ty-l6\*T\_RBtoD.coeff(1,0),Target\_tz-l6\*T\_RBtoD.coeff(2,0),&des\_sphe);

k->CarteToSphe(T\_RBtoEE.coeff(0,3)-l6\*T\_RBtoEE.coeff(0,0),T\_RBtoEE.coeff(1,3)-l6\*T\_RBtoEE.coeff(1,0),T\_RBtoEE.coeff(2,3)-l6\*T\_RBtoEE.coeff(2,0),&esti\_sphe);

double err\_r,err\_phid,err\_thetad;

err\_r=des\_sphe.R-esti\_sphe.R;

err\_phid=des\_sphe.phid-esti\_sphe.phid;

err\_thetad=des\_sphe.thetad-esti\_sphe.thetad;

des\_sphe.R = des\_sphe.R + 1 \* err\_r;

des\_sphe.phid = des\_sphe.phid + 1/360 \* err\_phid;

des\_sphe.thetad = des\_sphe.thetad + 1 \* err\_thetad;

Pos pos;

k->SpheToCarte(des\_sphe.R,des\_sphe.phid,des\_sphe.thetad,T\_RBtoD.coeff(0,0),T\_RBtoD.coeff(1,0),T\_RBtoD.coeff(2,0),&pos);

PoseD.x = pos.x;

PoseD.y = pos.y;

PoseD.z = pos.z;

\*/

/////////////////////

//cout << "#Spherical error# "<<endl<<" r: "<<err\_r << ", phid: "<<err\_phid << ", thetad: "<< err\_thetad<<endl;

/////////////////////

// --------------------------------------------- //

pTimer->diff\_target = sqrt((pTimer->bef\_Target\_tx - Target\_tx)\*(pTimer->bef\_Target\_tx - Target\_tx)+(pTimer->bef\_Target\_ty - Target\_ty)\*(pTimer->bef\_Target\_ty - Target\_ty)+(pTimer->bef\_Target\_tz - Target\_tz)\*(pTimer->bef\_Target\_tz - Target\_tz));

pTimer->bef\_Target\_tx = Target\_tx; pTimer->bef\_Target\_ty = Target\_ty; pTimer->bef\_Target\_tz = Target\_tz;

Pose PoseD; k->GetEuler(&T\_RBtoD,&PoseD);

Ikjoint InvK; k->ik(&PoseD,&InvK);

if( timerCount%1000 == 0 )

{

cout << "-----------------------------------------------------------"<<endl;

cout << "Desired target w.r.t RB coordinate : "<< endl << T\_RBtoD << endl << endl;

cout << "Orientation(Euler Angle, z-y-x ) : "<< PoseD.rz<< " " <<PoseD.ry <<" " <<PoseD.rx<< endl <<endl;

cout << "Position(x y z) :" << PoseD.x<<" " << PoseD.y<<" " << PoseD.z <<endl;

cout << "-----------------------------------------------------------"<<endl;

cout << "# Joint Value #" <<endl;

cout << InvK.InvJoint[0] << " , "<< InvK.InvJoint[1] << " , "<<InvK.InvJoint[2] << " , "<< InvK.InvJoint[3] << " , "<< InvK.InvJoint[4]<< " , " << InvK.InvJoint[5] << endl<< endl;

}

if( timerCount%100 == 0 )

{

cout << "-----------------------------------------------------------"<<endl;

cout << "#Position error# "<<endl<<" x: "<<err\_p.coeff(0) << ", y: "<<err\_p.coeff(1) << ", z: "<< err\_p.coeff(2)<<endl;

cout<< " Norm :" << err\_p.norm() << "[mm]"<<endl<<endl<<endl;

}

pTimer->m\_error = err\_p.norm();

pTimer->m\_Orierror = err\_o.norm();

if(bTargetFlag) // bTargetFlag : 네비게이션 시스템에서 전달받음

{

int i;

if( (30 < pTimer->m\_error)&&( pTimer->m\_error < 150 ) )

{

if(InvK.bIKFlag)

{

for(i=1;i<7;i++)

{

if(i == 3)

{

m->Move(MaxonHandle,i,InvK.InvJoint[i-1], err\_p.norm() \* 40 , err\_p.norm() \* 10 , 1);

continue;

}

m->Move(MaxonHandle,i,InvK.InvJoint[i-1], err\_p.norm() \* 10 , err\_p.norm() \* 10 , 1);

}//for문 종료

}

}

else if((5 < pTimer->m\_error)&&( pTimer->m\_error< 30.1))

{

//pTimer->OnBnClickedrobotbaseset();

if(InvK.bIKFlag)

{

for(i=1;i<7;i++)

{

if(i == 3)

{

m->Move(MaxonHandle,i,InvK.InvJoint[i-1], err\_p.norm() \* 20 , err\_p.norm() \* 30 , 1);

continue;

}

m->Move(MaxonHandle,i,InvK.InvJoint[i-1], err\_p.norm() \* 10 , err\_p.norm() \* 20 , 1);

}//for문 종료

}

}

else if((1.3 < pTimer->m\_error)&&( pTimer->m\_error< 5.1))

{

//T\_RBtoD.coeffRef(0,3) = Kp\*T\_RBtoD.coeff(0,3) + Ki\*err\_p.coeff(0) - Kd\*diff\_px;

//T\_RBtoD.coeffRef(1,3) = Kp\*T\_RBtoD.coeff(1,3) + Ki\*err\_p.coeff(1) - Kd\*diff\_py;

//T\_RBtoD.coeffRef(2,3) = Kp\*T\_RBtoD.coeff(2,3) + Ki\_Z\_axis\*err\_p.coeff(2) - Kd\*diff\_pz;

//Pose PoseD; k->GetEuler(&T\_RBtoD,&PoseD);

//Ikjoint InvK; k->ik(&PoseD,&InvK);

if(InvK.bIKFlag)

{

for(i=1;i<7;i++)

{

if(i == 3)

{

m->Move(MaxonHandle,i,InvK.InvJoint[i-1], err\_p.norm() \* 20 , err\_p.norm() \* 30 , 1);

continue;

}

m->Move(MaxonHandle,i,InvK.InvJoint[i-1], err\_p.norm() \* 10 , err\_p.norm() \* 20 , 1);

}//for문 종료

}

}

else

{

for(i=1;i<7;i++)

{

m->Halt(MaxonHandle,i);

}

}

/\*

if(pTimer->diff\_target > 1) // 마커 값이 살짝 노이즈가 생겨서

{

if(InvK.bIKFlag)

{for(int i=1;i<7;i++)

{m->Move(MaxonHandle,i,InvK.InvJoint[i-1],1);}}

}

else if(pTimer->m\_error > 30.0)

{

if(InvK.bIKFlag)

{for(int i=1;i<7;i++)

{m->Move(MaxonHandle,i,InvK.InvJoint[i-1],1);}

}

}

else if( (1.0 < pTimer->m\_error)&&( pTimer->m\_error< 30.0) )

{

if(InvK.bIKFlag)

{

for(int i=1;i<7;i++)

{

if(i == 3)

{

m->Move(MaxonHandle,i,InvK.InvJoint[i-1], err\_p.norm() \* 50 , err\_p.norm() \* 60 , 1);

continue;

}

m->Move(MaxonHandle,i,InvK.InvJoint[i-1], err\_p.norm() \* 10 , err\_p.norm() \* 40 , 1);

}//for문 종료

}

} // diff\_target 조건문 종료 \*/

//else

if( timerCount%100 == 0 )

{cout << "-----------------------------------------------------------"<<endl<<endl;}

}//bTargetFlag 조건문 종료

timerCount++;

delete k,m;

}