```
function [methodinfo,structs,enuminfo,ThunkLibName]=remoteApiProto
%REMOTEAPIPROTO Create structures to define interfaces found in 'extApi'.

%This function was generated by loadlibrary.m parser version 1.1.6.32
%perl options:'extApi.i -outfile=remoteApiProto.m'
ival={cell(1,0)}; % change 0 to the actual number of functions to preallocate the data.
structs=[];enuminfo=[];fcnNum=1;
fcns=struct('name',ival,'calltype',ival,'LHS',ival,'RHS',ival,'alias',ival);
ThunkLibName=[];

% extern  simxInt mtlb_simxSetJointPosition ( simxInt clientID , simxInt jointHandle , simxFloat*
position , simxInt operationMode );
fcns.name{fcnNum}='mtlb_simxSetJointPosition'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxSetJointTargetVelocity ( simxInt clientID , simxInt jointHandle ,
simxFloat* targetVelocity , simxInt operationMode );
fcns.name{fcnNum}='mtlb_simxSetJointTargetVelocity'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxSetJointTargetPosition ( simxInt clientID , simxInt jointHandle ,
simxFloat* targetPosition , simxInt operationMode );
fcns.name{fcnNum}='mtlb_simxSetJointTargetPosition'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxSetJointForce ( simxInt clientID , simxInt jointHandle , simxFloat*
force , simxInt operationMode );
fcns.name{fcnNum}='mtlb_simxSetJointForce'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxSetJointMaxForce ( simxInt clientID , simxInt jointHandle ,
simxFloat* force , simxInt operationMode );
fcns.name{fcnNum}='mtlb_simxSetJointMaxForce'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxSetFloatSignal ( simxInt clientID , const simxChar * signalName ,
simxFloat* signalValue , simxInt operationMode );
fcns.name{fcnNum}='mtlb_simxSetFloatSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxSetObjectFloatParameter ( simxInt clientID , simxInt objectHandle ,
simxInt parameterID , simxFloat* parameterValue , simxInt operationMode );
fcns.name{fcnNum}='mtlb_simxSetObjectFloatParameter'; fcns.calltype{fcnNum}='cdecl';
```

```
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxSetFloatingParameter ( simxInt clientID , simxInt paramIdentifier ,
simxFloat* paramValue , simxInt operationMode );
fcns.name{fcnNum}='mtlb_simxSetFloatingParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxCreateDummy ( simxInt clientID , simxFloat* size , const simxUChar
* colors , simxInt * objectHandle , simxInt operationMode );
fcns.name{fcnNum}='mtlb_simxCreateDummy'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'singlePtr', 'uint8Ptr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;

% extern  simxInt mtlb_simxReadProximitySensor(simxInt*
clientIDandSensorHandle,simxUChar* detectionState,simxFloat* detectedPoint,simxInt*
detectedObjectHandle,simxFloat* detectedSurfaceNormalVector,simxInt operationMode);
fcns.name{fcnNum}='mtlb_simxReadProximitySensor'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32Ptr', 'uint8Ptr', 'singlePtr', 'int32Ptr',
'singlePtr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxAuxiliaryConsoleOpen(simxInt*
clientIDandMaxLinesAndModeAndPositionAndSize,const simxChar* title,simxFloat*
textColor,simxFloat* backgroundColor,simxInt* consoleHandle,simxInt operationMode);
fcns.name{fcnNum}='mtlb_simxAuxiliaryConsoleOpen'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32Ptr', 'int8Ptr', 'singlePtr', 'singlePtr',
'int32Ptr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxDisplayDialog(simxInt*
clientIDandDlgTypeAndOperationMode,const simxChar* titleText,const simxChar*
mainText,const simxChar* initialText,const simxFloat* titleColorsAndDlgColors,simxInt*
dialogHandleAndUiHandle);
fcns.name{fcnNum}='mtlb_simxDisplayDialog'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32Ptr', 'int8Ptr', 'int8Ptr','int8Ptr', 'singlePtr',
'int32Ptr'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxQuery(simxInt* clientIDandSignalLengthAndTimeOutInMs,const
simxChar* signalName,const simxUChar* signalValue,const simxChar*
retSignalName,simxUChar** retSignalValue,simxInt* retSignalLength);
fcns.name{fcnNum}='mtlb_simxQuery'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int32Ptr', 'int8Ptr', 'uint8Ptr', 'int8Ptr', 'uint8PtrPtr',
'int32Ptr'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxGetObjectGroupData(simxInt*
clientIDandObjectTypeAndDataTypeAndOperationMode,simxInt*
handlesCountAndIntDataCountAndFloatDataCountAndStringDataCount,simxInt**
handles,simxInt** intData,simxFloat** floatData,simxChar** stringData);
fcns.name{fcnNum}='mtlb_simxGetObjectGroupData'; fcns.calltype{fcnNum}='cdecl';
```

```
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32Ptr', 'int32Ptr', 'int32PtrPtr', 'int32PtrPtr',
'singlePtrPtr', 'int8PtrPtr'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxCallScriptFunction_a(const simxInt* variousIntsIn,const simxChar*
scriptDescriptionAndFunctionName,const simxInt* inInt,const simxFloat* inFloat,const
simxChar* inString,const simxUChar* inBuffer);
fcns.name{fcnNum}='mtlb_simxCallScriptFunction_a'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32Ptr', 'int8Ptr', 'int32Ptr', 'singlePtr', 'int8Ptr',
'uint8Ptr'};fcnNum=fcnNum+1;
% extern  simxInt mtlb_simxCallScriptFunction_b(simxInt clientID,simxInt*
variousIntsOut,simxInt** outInt,simxFloat** outFloat,simxChar** outString,simxUChar**
outBuffer);
fcns.name{fcnNum}='mtlb_simxCallScriptFunction_b'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32Ptr', 'int32PtrPtr', 'singlePtrPtr',
'int8PtrPtr', 'uint8PtrPtr'};fcnNum=fcnNum+1;



% extern  simxInt simxGetJointPosition ( simxInt clientID , simxInt jointHandle , simxFloat *
position , simxInt operationMode );
fcns.name{fcnNum}='simxGetJointPosition'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetJointMatrix ( simxInt clientID , simxInt jointHandle , simxFloat * matrix
, simxInt operationMode );
fcns.name{fcnNum}='simxGetJointMatrix'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetSphericalJointMatrix ( simxInt clientID , simxInt jointHandle , simxFloat
* matrix , simxInt operationMode );
fcns.name{fcnNum}='simxSetSphericalJointMatrix'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetJointForce ( simxInt clientID , simxInt jointHandle , simxFloat * force ,
simxInt operationMode );
fcns.name{fcnNum}='simxGetJointForce'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetJointMaxForce ( simxInt clientID , simxInt jointHandle , simxFloat *
force , simxInt operationMode );
fcns.name{fcnNum}='simxGetJointMaxForce'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxReadForceSensor ( simxInt clientID , simxInt forceSensorHandle ,
```

```
simxUChar * state , simxFloat * forceVector , simxFloat * torqueVector , simxInt operationMode
);
fcns.name{fcnNum}='simxReadForceSensor'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'uint8Ptr', 'singlePtr', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxBreakForceSensor ( simxInt clientID , simxInt forceSensorHandle ,
simxInt operationMode );
fcns.name{fcnNum}='simxBreakForceSensor'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxReadVisionSensor ( simxInt clientID , simxInt sensorHandle , simxUChar
* detectionState , simxFloat ** auxValues , simxInt ** auxValuesCount , simxInt operationMode
);
fcns.name{fcnNum}='simxReadVisionSensor'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'uint8Ptr', 'singlePtrPtr',
'int32PtrPtr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetObjectHandle ( simxInt clientID , const simxChar * objectName ,
simxInt * handle , simxInt operationMode );
fcns.name{fcnNum}='simxGetObjectHandle'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetVisionSensorImage ( simxInt clientID , simxInt sensorHandle , simxInt
* resolution , simxUChar ** image , simxUChar options , simxInt operationMode );
fcns.name{fcnNum}='simxGetVisionSensorImage'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32Ptr', 'uint8PtrPtr', 'uint8',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetVisionSensorImage ( simxInt clientID , simxInt sensorHandle ,
simxUChar * image , simxInt bufferSize , simxUChar options , simxInt operationMode );
fcns.name{fcnNum}='simxSetVisionSensorImage'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'uint8Ptr', 'int32', 'uint8',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetVisionSensorDepthBuffer ( simxInt clientID , simxInt sensorHandle ,
simxInt * resolution , simxFloat ** buffer , simxInt operationMode );
fcns.name{fcnNum}='simxGetVisionSensorDepthBuffer'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32Ptr', 'singlePtrPtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetObjectChild ( simxInt clientID , simxInt parentObjectHandle , simxInt
childIndex , simxInt * childObjectHandle , simxInt operationMode );
fcns.name{fcnNum}='simxGetObjectChild'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetObjectParent ( simxInt clientID , simxInt childObjectHandle , simxInt *
parentObjectHandle , simxInt operationMode );
fcns.name{fcnNum}='simxGetObjectParent'; fcns.calltype{fcnNum}='cdecl';
```

```
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxLoadModel ( simxInt clientID , const simxChar * modelPathAndName ,
simxUChar options , simxInt * baseHandle , simxInt operationMode );
fcns.name{fcnNum}='simxLoadModel'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'uint8', 'int32Ptr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxLoadUI ( simxInt clientID , const simxChar * uiPathAndName ,
simxUChar options , simxInt * count , simxInt ** uiHandles , simxInt operationMode );
fcns.name{fcnNum}='simxLoadUI'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'uint8', 'int32Ptr', 'int32PtrPtr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxLoadScene ( simxInt clientID , const simxChar * scenePathAndName ,
simxUChar options , simxInt operationMode );
fcns.name{fcnNum}='simxLoadScene'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'uint8', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxStartSimulation ( simxInt clientID , simxInt operationMode );
fcns.name{fcnNum}='simxStartSimulation'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxPauseSimulation ( simxInt clientID , simxInt operationMode );
fcns.name{fcnNum}='simxPauseSimulation'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxStopSimulation ( simxInt clientID , simxInt operationMode );
fcns.name{fcnNum}='simxStopSimulation'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetUIHandle ( simxInt clientID , const simxChar * uiName , simxInt *
handle , simxInt operationMode );
fcns.name{fcnNum}='simxGetUIHandle'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetUISlider ( simxInt clientID , simxInt uiHandle , simxInt uiButtonID ,
simxInt * position , simxInt operationMode );
fcns.name{fcnNum}='simxGetUISlider'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'int32Ptr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetUISlider ( simxInt clientID , simxInt uiHandle , simxInt uiButtonID ,
simxInt position , simxInt operationMode );
fcns.name{fcnNum}='simxSetUISlider'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetUIEventButton ( simxInt clientID , simxInt uiHandle , simxInt *
uiEventButtonID , simxInt * auxValues , simxInt operationMode );
fcns.name{fcnNum}='simxGetUIEventButton'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32Ptr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetUIButtonProperty ( simxInt clientID , simxInt uiHandle , simxInt
uiButtonID , simxInt * prop , simxInt operationMode );
```

```
fcns.name{fcnNum}='simxGetUIButtonProperty'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetUIButtonProperty ( simxInt clientID , simxInt uiHandle , simxInt
uiButtonID , simxInt prop , simxInt operationMode );
fcns.name{fcnNum}='simxSetUIButtonProperty'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'int32',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxAddStatusbarMessage ( simxInt clientID , const simxChar * message ,
simxInt operationMode );
fcns.name{fcnNum}='simxAddStatusbarMessage'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxAuxiliaryConsoleClose ( simxInt clientID , simxInt consoleHandle , simxInt
operationMode );
fcns.name{fcnNum}='simxAuxiliaryConsoleClose'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxAuxiliaryConsolePrint ( simxInt clientID , simxInt consoleHandle , const
simxChar * txt , simxInt operationMode );
fcns.name{fcnNum}='simxAuxiliaryConsolePrint'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int8Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxAuxiliaryConsoleShow ( simxInt clientID , simxInt consoleHandle ,
simxUChar showState , simxInt operationMode );
fcns.name{fcnNum}='simxAuxiliaryConsoleShow'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'uint8', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetObjectOrientation ( simxInt clientID , simxInt objectHandle , simxInt
relativeToObjectHandle , simxFloat * eulerAngles , simxInt operationMode );
fcns.name{fcnNum}='simxGetObjectOrientation'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetObjectQuaternion ( simxInt clientID , simxInt objectHandle , simxInt
relativeToObjectHandle , simxFloat * eulerAngles , simxInt operationMode );
fcns.name{fcnNum}='simxGetObjectQuaternion'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetObjectPosition ( simxInt clientID , simxInt objectHandle , simxInt
relativeToObjectHandle , simxFloat * position , simxInt operationMode );
fcns.name{fcnNum}='simxGetObjectPosition'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetObjectOrientation ( simxInt clientID , simxInt objectHandle , simxInt
relativeToObjectHandle , const simxFloat * eulerAngles , simxInt operationMode );
fcns.name{fcnNum}='simxSetObjectOrientation'; fcns.calltype{fcnNum}='cdecl';
```

```
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetObjectQuaternion ( simxInt clientID , simxInt objectHandle , simxInt
relativeToObjectHandle , const simxFloat * eulerAngles , simxInt operationMode );
fcns.name{fcnNum}='simxSetObjectQuaternion'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetObjectPosition ( simxInt clientID , simxInt objectHandle , simxInt
relativeToObjectHandle , const simxFloat * position , simxInt operationMode );
fcns.name{fcnNum}='simxSetObjectPosition'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetObjectParent ( simxInt clientID , simxInt objectHandle , simxInt
parentObject , simxUChar keepInPlace , simxInt operationMode );
fcns.name{fcnNum}='simxSetObjectParent'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'uint8',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetUIButtonLabel ( simxInt clientID , simxInt uiHandle , simxInt
uiButtonID , const simxChar * upStateLabel , const simxChar * downStateLabel , simxInt
operationMode );
fcns.name{fcnNum}='simxSetUIButtonLabel'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'int8Ptr', 'int8Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetLastErrors ( simxInt clientID , simxInt * errorCnt , simxChar **
errorStrings , simxInt operationMode );
fcns.name{fcnNum}='simxGetLastErrors'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32Ptr', 'int8PtrPtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetArrayParameter ( simxInt clientID , simxInt paramIdentifier , simxFloat
* paramValues , simxInt operationMode );
fcns.name{fcnNum}='simxGetArrayParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetArrayParameter ( simxInt clientID , simxInt paramIdentifier , const
simxFloat * paramValues , simxInt operationMode );
fcns.name{fcnNum}='simxSetArrayParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetBooleanParameter ( simxInt clientID , simxInt paramIdentifier ,
simxUChar * paramValue , simxInt operationMode );
fcns.name{fcnNum}='simxGetBooleanParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'uint8Ptr',
'int32'};fcnNum=fcnNum+1;
```

```
% extern  simxInt simxSetBooleanParameter ( simxInt clientID , simxInt paramIdentifier ,
simxUChar paramValue , simxInt operationMode );
fcns.name{fcnNum}='simxSetBooleanParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'uint8', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetIntegerParameter ( simxInt clientID , simxInt paramIdentifier , simxInt
* paramValue , simxInt operationMode );
fcns.name{fcnNum}='simxGetIntegerParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetIntegerParameter ( simxInt clientID , simxInt paramIdentifier , simxInt
paramValue , simxInt operationMode );
fcns.name{fcnNum}='simxSetIntegerParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetFloatingParameter ( simxInt clientID , simxInt paramIdentifier ,
simxFloat * paramValue , simxInt operationMode );
fcns.name{fcnNum}='simxGetFloatingParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetStringParameter ( simxInt clientID , simxInt paramIdentifier , simxChar
** paramValue , simxInt operationMode );
fcns.name{fcnNum}='simxGetStringParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int8PtrPtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetCollisionHandle ( simxInt clientID , const simxChar *
collisionObjectName , simxInt * handle , simxInt operationMode );
fcns.name{fcnNum}='simxGetCollisionHandle'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetDistanceHandle ( simxInt clientID , const simxChar *
distanceObjectName , simxInt * handle , simxInt operationMode );
fcns.name{fcnNum}='simxGetDistanceHandle'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetCollectionHandle ( simxInt clientID , const simxChar * collectionName
, simxInt * handle , simxInt operationMode );
fcns.name{fcnNum}='simxGetCollectionHandle'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxReadCollision ( simxInt clientID , simxInt collisionObjectHandle ,
simxUChar * collisionState , simxInt operationMode );
fcns.name{fcnNum}='simxReadCollision'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'uint8Ptr',
'int32'};fcnNum=fcnNum+1;
```

```
% extern  simxInt simxReadDistance ( simxInt clientID , simxInt distanceObjectHandle ,
simxFloat * minimumDistance , simxInt operationMode );
fcns.name{fcnNum}='simxReadDistance'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxRemoveObject ( simxInt clientID , simxInt objectHandle , simxInt
operationMode );
fcns.name{fcnNum}='simxRemoveObject'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxRemoveModel ( simxInt clientID , simxInt objectHandle , simxInt
operationMode );
fcns.name{fcnNum}='simxRemoveModel'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxRemoveUI ( simxInt clientID , simxInt uiHandle , simxInt operationMode );
fcns.name{fcnNum}='simxRemoveUI'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int32', 'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxCloseScene ( simxInt clientID , simxInt operationMode );
fcns.name{fcnNum}='simxCloseScene'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetObjects ( simxInt clientID , simxInt objectType , simxInt * objectCount
, simxInt ** objectHandles , simxInt operationMode );
fcns.name{fcnNum}='simxGetObjects'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int32', 'int32', 'int32Ptr', 'int32PtrPtr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxEndDialog ( simxInt clientID , simxInt dialogHandle , simxInt
operationMode );
fcns.name{fcnNum}='simxEndDialog'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int32', 'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetDialogInput ( simxInt clientID , simxInt dialogHandle , simxChar **
inputText , simxInt operationMode );
fcns.name{fcnNum}='simxGetDialogInput'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int8PtrPtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetDialogResult ( simxInt clientID , simxInt dialogHandle , simxInt * result
, simxInt operationMode );
fcns.name{fcnNum}='simxGetDialogResult'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxCopyPasteObjects ( simxInt clientID , const simxInt * objectHandles ,
simxInt objectCount , simxInt ** newObjectHandles , simxInt * newObjectCount , simxInt
operationMode );
fcns.name{fcnNum}='simxCopyPasteObjects'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32Ptr', 'int32', 'int32PtrPtr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
```

```
% extern  simxInt simxGetObjectSelection ( simxInt clientID , simxInt ** objectHandles , simxInt
* objectCount , simxInt operationMode );
fcns.name{fcnNum}='simxGetObjectSelection'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32PtrPtr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetObjectSelection ( simxInt clientID , const simxInt * objectHandles ,
simxInt objectCount , simxInt operationMode );
fcns.name{fcnNum}='simxSetObjectSelection'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32Ptr', 'int32',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxClearFloatSignal ( simxInt clientID , const simxChar * signalName ,
simxInt operationMode );
fcns.name{fcnNum}='simxClearFloatSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxClearIntegerSignal ( simxInt clientID , const simxChar * signalName ,
simxInt operationMode );
fcns.name{fcnNum}='simxClearIntegerSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxClearStringSignal ( simxInt clientID , const simxChar * signalName ,
simxInt operationMode );
fcns.name{fcnNum}='simxClearStringSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetFloatSignal ( simxInt clientID , const simxChar * signalName ,
simxFloat * signalValue , simxInt operationMode );
fcns.name{fcnNum}='simxGetFloatSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetIntegerSignal ( simxInt clientID , const simxChar * signalName ,
simxInt * signalValue , simxInt operationMode );
fcns.name{fcnNum}='simxGetIntegerSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetStringSignal ( simxInt clientID , const simxChar * signalName ,
simxUChar ** signalValue , simxInt * signalLength , simxInt operationMode );
fcns.name{fcnNum}='simxGetStringSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'uint8PtrPtr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetAndClearStringSignal ( simxInt clientID , const simxChar *
signalName , simxUChar ** signalValue , simxInt * signalLength , simxInt operationMode );
fcns.name{fcnNum}='simxGetAndClearStringSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'uint8PtrPtr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxReadStringStream ( simxInt clientID , const simxChar * signalName ,
```

simxUChar ** signalValue , simxInt * signalLength , simxInt operationMode);
fcns.name{fcnNum}='simxReadStringStream'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'uint8PtrPtr', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern simxInt simxSetIntegerSignal (simxInt clientID , const simxChar * signalName ,
simxInt signalValue , simxInt operationMode);
fcns.name{fcnNum}='simxSetIntegerSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32',
'int32'};fcnNum=fcnNum+1;
% extern simxInt simxSetStringSignal (simxInt clientID , const simxChar * signalName , const
simxUChar * signalValue , simxInt signalLength , simxInt operationMode);
fcns.name{fcnNum}='simxSetStringSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'uint8Ptr', 'int32',
'int32'};fcnNum=fcnNum+1;
% extern simxInt simxAppendStringSignal (simxInt clientID , const simxChar * signalName ,
const simxUChar * signalValue , simxInt signalLength , simxInt operationMode);
fcns.name{fcnNum}='simxAppendStringSignal'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'uint8Ptr', 'int32',
'int32'};fcnNum=fcnNum+1;
% extern simxInt simxWriteStringStream (simxInt clientID , const simxChar * signalName ,
const simxUChar * signalValue , simxInt signalLength , simxInt operationMode);
fcns.name{fcnNum}='simxWriteStringStream'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'uint8Ptr', 'int32',
'int32'};fcnNum=fcnNum+1;
% extern simxInt simxGetObjectFloatParameter (simxInt clientID , simxInt objectHandle ,
simxInt parameterID , simxFloat * parameterValue , simxInt operationMode);
fcns.name{fcnNum}='simxGetObjectFloatParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern simxInt simxGetObjectIntParameter (simxInt clientID , simxInt objectHandle , simxInt
parameterID , simxInt * parameterValue , simxInt operationMode);
fcns.name{fcnNum}='simxGetObjectIntParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'int32Ptr',
'int32'};fcnNum=fcnNum+1;
% extern simxInt simxSetObjectIntParameter (simxInt clientID , simxInt objectHandle , simxInt
parameterID , simxInt parameterValue , simxInt operationMode);
fcns.name{fcnNum}='simxSetObjectIntParameter'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'int32',
'int32'};fcnNum=fcnNum+1;
% extern simxInt simxGetModelProperty (simxInt clientID , simxInt objectHandle , simxInt *
prop , simxInt operationMode);
fcns.name{fcnNum}='simxGetModelProperty'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32Ptr',

```
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSetModelProperty ( simxInt clientID , simxInt objectHandle , simxInt prop
, simxInt operationMode );
fcns.name{fcnNum}='simxSetModelProperty'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetObjectVelocity ( simxInt clientID , simxInt objectHandle , simxFloat *
linearVelocity , simxFloat * angularVelocity , simxInt operationMode );
fcns.name{fcnNum}='simxGetObjectVelocity'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'singlePtr', 'singlePtr',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxStart ( const simxChar * connectionAddress , simxInt connectionPort ,
simxUChar waitUntilConnected , simxUChar doNotReconnectOnceDisconnected , simxInt
timeOutInMs , simxInt commThreadCycleInMs );
fcns.name{fcnNum}='simxStart'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int8Ptr', 'int32', 'uint8', 'uint8', 'int32', 'int32'};fcnNum=fcnNum+1;
% extern  simxVoid simxFinish ( simxInt clientID );
fcns.name{fcnNum}='simxFinish'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}=[];
fcns.RHS{fcnNum}={'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxGetPingTime ( simxInt clientID , simxInt * pingTime );
fcns.name{fcnNum}='simxGetPingTime'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32Ptr'};fcnNum=fcnNum+1;
% extern  simxInt simxGetLastCmdTime ( simxInt clientID );
fcns.name{fcnNum}='simxGetLastCmdTime'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSynchronousTrigger ( simxInt clientID );
fcns.name{fcnNum}='simxSynchronousTrigger'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxSynchronous ( simxInt clientID , simxUChar enable );
fcns.name{fcnNum}='simxSynchronous'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'uint8'};fcnNum=fcnNum+1;
% extern  simxInt simxPauseCommunication ( simxInt clientID , simxUChar pause );
fcns.name{fcnNum}='simxPauseCommunication'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'uint8'};fcnNum=fcnNum+1;
% extern  simxInt simxGetInMessageInfo ( simxInt clientID , simxInt infoType , simxInt * info );
fcns.name{fcnNum}='simxGetInMessageInfo'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32Ptr'};fcnNum=fcnNum+1;
% extern  simxInt simxGetOutMessageInfo ( simxInt clientID , simxInt infoType , simxInt * info );
fcns.name{fcnNum}='simxGetOutMessageInfo'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int32', 'int32Ptr'};fcnNum=fcnNum+1;
% extern  simxInt simxGetConnectionId ( simxInt clientID );
fcns.name{fcnNum}='simxGetConnectionId'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32'};fcnNum=fcnNum+1;
% extern  simxUChar * simxCreateBuffer ( simxInt bufferSize );
```

```
fcns.name{fcnNum}='simxCreateBuffer'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='uint8Ptr'; fcns.RHS{fcnNum}={'int32'};fcnNum=fcnNum+1;
% extern  simxVoid simxReleaseBuffer ( simxUChar * buffer );
fcns.name{fcnNum}='simxReleaseBuffer'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}=[];
fcns.RHS{fcnNum}={'uint8Ptr'};fcnNum=fcnNum+1;
% extern  simxInt simxTransferFile ( simxInt clientID , const simxChar * filePathAndName ,
const simxChar * fileName_serverSide , simxInt timeOut , simxInt operationMode );
fcns.name{fcnNum}='simxTransferFile'; fcns.calltype{fcnNum}='cdecl';
fcns.LHS{fcnNum}='int32'; fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int8Ptr', 'int32',
'int32'};fcnNum=fcnNum+1;
% extern  simxInt simxEraseFile ( simxInt clientID , const simxChar * fileName_serverSide ,
simxInt operationMode );
fcns.name{fcnNum}='simxEraseFile'; fcns.calltype{fcnNum}='cdecl'; fcns.LHS{fcnNum}='int32';
fcns.RHS{fcnNum}={'int32', 'int8Ptr', 'int32'};fcnNum=fcnNum+1;
methodinfo=fcns;
```