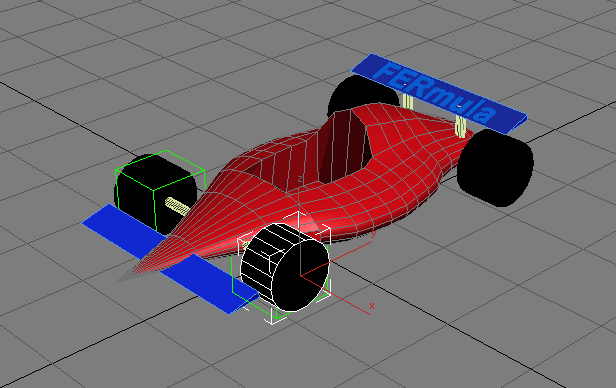
Vrlo važan dio tehničke demonstracije proširene stvarnosti zahtjeva implementaciju interakcije u stvarnom vremenu. U tu svrhu ostvareno je pokretanje vozila FERmula zadavanjem naredbi preko tipkovnice.

U nastavku opisan je način na koji je implementirano pokretanje FERmule: ubrzavanje, usporavanje, skretanje FERmule, te zakretanje kotača.

1) ZAKRETANJE KOTAČA

Prilikom skretanja FERmule njezini prednji kotači pomiču se u odgovarajuću stranu, tj. rotiraju se za kut od 30° . Prilikom ostvarenja ove funkcionalnosti koristimo neke gotove funkcije. Funkcija *FindNodeByName* izdvaja nam dio modela čije ime predajemo funkciji kao argument. Nad pronađenim dijelom modela dalje je moguće vršiti transformacije. Da bi uspješno izdvojili dijelove modela, tj. kotače, potrebno ih je nazvati fiksnim imenima. U našem slučaju prednji kotači su nazvani *kotacPL* i *kotacPD*. Dodatno, funkcija *FindNodeByName* ograničava broj roditelja dijela modela koji se izdvaja. Iz tog razloga kotačima je pridodan točno jedan roditelj, nevidljivi objekt (tzv. *Dummy*)(slika 1.)*.*

**„Dummy“**

**kotacPD**

**kotacPL**

Slika 1. Model FERmule i imena kotača

Pritiskom na tipke za skretanje (KEY DOWN: left, right) poziva se odgovarajuća funkcija za zakretanje kotača (*okreniLijevo*, *okreniDesno*). Ukoliko su kotači već zakrenuti funkcije *okreniLijevo* i *okreniDesno* ne zakreću dodatno kotače, čime se sprečava beskonačna rotacija. Prilikom otpuštanja odgovarajućih tipki za skretanje (KEY UP: left, right) kotači se vraćaju u početni položaj, tj. rotiraju se za kut -30°.

case osgGA::GUIEventAdapter::KEY\_Right:

{

voziloInputDeviceState->rotDReq = false;

v->**okreniLijevo(false)**;

return false;

}

void okreniLijevo(bool t)

{

if (t) okrenutL = 1;

else okrenutD = 0;

osg::Node\* lijeviKotac = FindNodeByName( Model, "kotacPL" );

osg::Node\* desniKotac = FindNodeByName( Model, "kotacPD" );

osg::MatrixTransform\* ltr = AddMatrixTransform(lijeviKotac);

osg::MatrixTransform\* dtr = AddMatrixTransform(desniKotac);

ltr->setMatrix(osg::Matrix::rotate(osg::inDegrees(30.0f),osg::Z\_AXIS));

dtr->setMatrix(osg::Matrix::rotate(osg::inDegrees(30.0f),osg::Z\_AXIS));

}

osg::Node\* FindNodeByName( osg::Node\* pNode, const std::string& sName )

{

if ( pNode->getName()==sName )

{

return pNode;

}

osg::Group\* pGroup = pNode->asGroup();

if ( pGroup )

{

for ( unsigned int i=0; i<pGroup->getNumChildren(); i++ )

{

osg::Node\* pFound = FindNodeByName( pGroup->getChild(i), sName );

if ( pFound )

{

return pFound;

}

}

}

return 0;

}