Etape 1

Dans le fichier **tele1.cpp**, on associe le tag laser à une fonction callback d'URBI.

```
// recuperation de la position des lasers
urbiTele.robotC->setCallback(callback(urbiTele,&ConnectAibo::returnvaluelasers),"totolasers");
urbiTele.robotC->setCallback(callback(urbiTele,&ConnectAibo::returnvaluelasers2),"totolasersanglemin");
urbiTele.robotC->setCallback(callback(urbiTele,&ConnectAibo::returnvaluelasers2),"totolasersanglemax");
urbiTele.robotC->setCallback(callback(urbiTele,&ConnectAibo::returnvaluelasers2),"totolasercount");
```

Etape 2

Dans le fichier **tele4.cpp** se trouve le code qui se lance quand on appuie sur les boutons relatifs au laser.

```
urbiTele.robotC->send("totolasersanglemin: laser.angleMin;");
urbiTele.robotC->send("totolasersanglemax: laser.angleMax;");
urbiTele.robotC->send("totolasercount: laser.count;");

urbiTele.robotC->send("stop totolasers;");
urbiTele.robotC->send("totolasers:loop {");
urbiTele.robotC->send("totolasers: laser.val");
urbiTele.robotC->send("},");
```

Etape 3

Le fichier **urbi_page5.cpp** contient les fonctions callbacks relatives au laser.

La fonction **returnvaluelasers2()** permet de récupérer les valeurs de **laser.angleMin**, **laser.angleMax** et **laser.count**.

```
QString tag = QString::fromLatin1(msg.tag.c_str());
double value = (double) msg.value->val;

 if( tag == "totolasercount")
    lasercount=value;
 else if( tag == "totolasersanglemin")
    laseranglemin=value;
 else if( tag == "totolasersanglemax")
    laseranglemax=value;
```

La fonction **returnvaluelasers()** permet de récupérer les valeurs de **laser.val**.
Les valeurs de **laser.val** sont stockées dans un tableau de double : **valuelasers**.
A l'aide de ces distances, on détermine la position de tous les points de mesure. On stocke ces données dans un tableau de coordonnées : **pointslaser**.

A ce stade, il ne reste plus qu'à afficher l'image du laser mais il n'est pas possible de le faire depuis un callback URBI car cela provoque une erreur QT. En conséquence, il faut le faire depuis un événement.

```
e1 = new QEvent((QEvent::Type)(1005));
QcoreApplication::postEvent(this,e1);
```

Etape 4

Dans le fichier urbi.cpp, on fait une fonction pour réceptionner l'événement.
On place les points dans une image et on affiche ensuite l'image.

```
bool ConnectAibo::event(QEvent *ev)
{
 if( ev->type() == (QEvent::Type)(1005)) // laser
 {

      imagelaserclean = imagelaserclean2; // on recupere l'image initiale

      QPainter painter;
      painter.begin(&imagelaserclean);

      // on fait 4 cercles
      painter.setPen(Qt::black);
      painter.drawEllipse ( 225, 225, 150, 150 ); // 2 metres
      painter.drawEllipse ( 150, 150, 300, 300 ); // 4 metres
      painter.drawEllipse ( 75, 75, 450, 450 ); // 6 metres
      painter.drawEllipse ( 0, 0, 600, 600 ); // 8 metres

      // on place les points du laser
      painter.setPen(Qt::blue);
      painter.drawConvexPolygon(pointslaser, lasercount);

      painter.end();

      pere->ui.label_26->setPixmap(QPixmap::fromImage(imagelaserclean));     // on affiche
 }

 return 1;
}
```