Follow-focus with Arduino

Christian Guerrero
Ludwig Maximilian Universität, München, Munich, Germany
christian.guerrero@campus.lmu.de

Arthur Moufounda
École Supérieure Des Technologies Industrielles Avancées, Bidart, France
a.moufounda@net.estia.fr

Alexander Schenker
Ludwig Maximilian Universität, München, Munich, Germany
alexander.schenker@campus.lmu.de

ABSTRACT

Author Keywords
Follow Focus; Movie Production; Remote Control; Film Set Automation; Arduino; Android;

MOTIVATION AND BACKGROUND

General Motivation

General Description of the Topic

In movie production, especially the part of *principal photography* is considered one of the most expensive ones. This is, amongst other reasons, due to the often high number of scene takes, that have to be shot. [7] Usually, manual control is required in order to zoom and to keep the focus on the desired object. In classical productions, the *focus puller* (or *first Assistant Camera*) is a dedicated position, whose job it is to keep the focus according to the cinematographers instructions. As there is no way to correct a falsely set focus in post-production, it is considered one of the hardest jobs on a film set. [10]

State of the Art in Follow Focus

This section gives a brief overview about a selection of currently existing technologies in the field of follow focus.

Andra Motion Focus

Andra Motion Focus is a commercial hardware system developed by Andra Motion Technologies Inc. It combines hardware-supported motion capturing with a remote controlled follow focus system. It is currently controllable optionally with an iPad or a dedicated hardware interface, called *The Arc*. Andra Motion Focus is designed for portability and efficiency, in order to reduce the amount of time needed and hence the shoots' cost. Due to the high price, it aims for professional and semi-professional film-makers. [5]

Lenzhound

Lenzhound is "A super-affordable wireless follow focus system for indie filmmakers." [3] It was originally featured on kickstarter by a company called Motion Dogs. The key idea of Lenzhound is, to have a simplistic, hardware- and remote-controlled Focus System. It is belt-driven and can be attached to almost any type of camera lens. Unlike Andra Motion Focus, Lenzhound addresses both amateur and independent film-makers.

Soffer Follow Focus

"Follow Focus" by Adi Soffer is an open source and Do-it-yourself (DIY) follow focus project from 2013. It is based on Arduino hardware and standard electronical components, which makes it relatively easy to reproduce. The source code can be found on GitHub. [9]

PROAIM Camera Follow Focus X1 (FF-X1)

The FF-X1 by PROAIM Camera is exemplarily for many similar products on the market. It is a purely manual gear-driven mechanism, that is supposed to "ensure smooth accurate emphasis over your subject" [4]. Usage is supported for a wide range of cameras.

Background

Vision

Considering the difficulties resulting from the precision required while shooting a film, and the often high number of takes, it is worth thinking about a holistic automated system. One that offers not only automated dolly shots, or remote controlled focus pulling, but one that includes everything from camera movement to image composition. As mentioned in the State of the Art Section, there is a wide range of systems available. However, they are often not affordable for independent film-makers, due to their high costs. Also, for most solutions, the benefit is limited to one single use-case, such as focus pulling.

Consequently, our vision is to create a DIY system, which combines features from severeal existing solutions.

Current Research at LMU

Previous and current research conducted by *Ludwig-Maximilian-Universität* (LMU) in Munich has come up with the development of a remote controlled camera slider [8] and further theoretical research on automated camera shots [6].

This paper covers the prototypical development of a remote controlled focus pulling device with the long-term objective to get inegrated into a system, which combines automated dolly shots, focus pulling and camera tilts.

SOLUTION

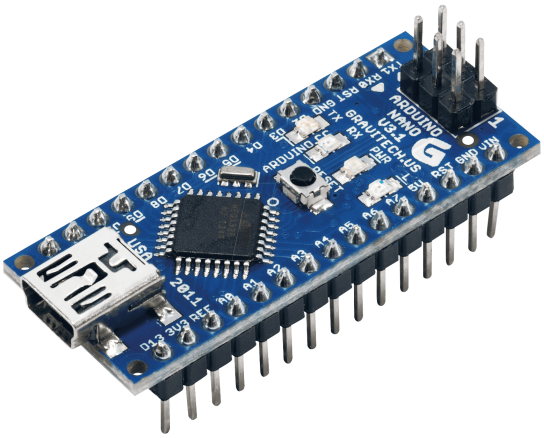
Our Follow Focus Approach

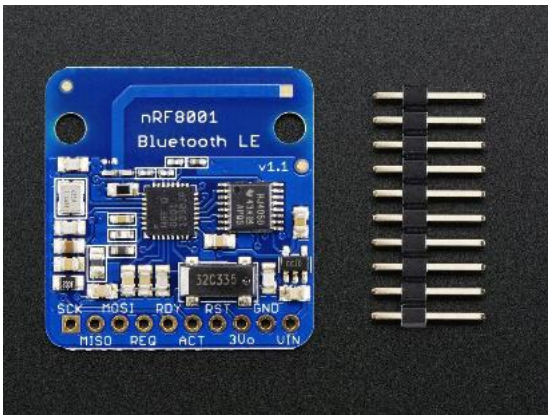
Our approach is based on the ideas of *DIY Follow Focus* and *Soffer Follow Focus*, combined with an app-based remote control interface.

Having analyzed the existing solutions from the *State of the Art* section, we came up with a list of requirements, that we impose on our prototype:

- The protypical device is adaptable to a wide range of camera types.
- Focus pulling can be done in realtime.
- The speed of focus change is adjustable.
- The lense's range is calibratable.
- Focus sequences are recordable and therefore easily repeatable.
- The system is controllable through an Android-based app interface.
- The communication between hardware and software happens via Bluetooth Low Energy (BLE).

Short BLE description here! TODO





Methods

To realize our aims we decided to develop an Android App for input. It sends a perpetual signal via BLE to a BLE module connected to an Arduino Nano. The Arduino processes the byte information and sends commands to a stepper driver module. The driver then controls the movement of a stepper motor.

Electrical Engineering

Top of all, we wanted our follow focus to be portable. Therefore the elements had to be small, low energy powered and easily embeddable. These challenges had to be tackled in the planning phase of the process. So at first we had to decide between different technologies in order to meet our aims:

- The remote controller would be a smart phone. Thus, anyone could easily control the follow focus device by downloading our application in the app store.
- For our requirements an Arduino Nano is more than sufficient. It is small and consumes only little.
- For the Wireless control we use Bluetooth LE because of its low energy, easy implementation and also because there is a large web community and many examples available online.
- A stepper motor is used because it works more precise and faster than a servo and has only small enertia. Also with stepper we always know the exact amount of rotation we have to apply.
- Having a motor is good. Adapting it to an electrical system is better. The stepper driver will protect the Arduino Nano from different electrical damages that may be caused by

the sudden increase of intensity caused by the motor. It also allows to turn in both directions.

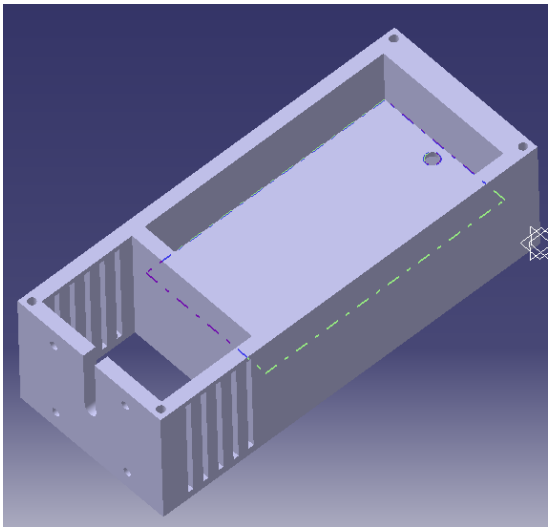
- The AccelStepper library for Arduino is very useful as it makes the API never to block and supports very slow speeds. Also it helps to easily apply acceleration.
- None of the electronic parts is passive. To make them all work, we need electric energy, but we also want our global system to be portable. Therefore we will use different sets of battery, because the Arduino and the step motor need to be powered separately, at least 5V and 12V respectively.

Listing 1 shows how the received BLE Serial date is being processed on Arduino side.

Listing 1. Processing of received BLE serial data

```
if (status == ACI_EVT_CONNECTED)
  {
    stepper.enableOutputs();
    stepper.run();
    if (BLEserial.available()>0)
    {
      digitalWrite (enablePin, LOW);
      previousMillis = millis();
      dataReceive = BLEserial.read();
      react(); //function to react to input
      stepper.run();
      stepper.moveTo(encoderValue);
    }
        ...

void react()
{
  if (dataReceive > 48 && dataReceive < 58)
//[1-9]-Key
  {
    speedValue = map(dataReceive, 49, 57,
        minSpeedValue, maxSpeedValue);
    stepper.setMaxSpeed(speedValue);
    Serial.println(speedValue);
  }
  else if (dataReceive == 43) //[+]-Key
  {
    encoderValue += stepSize;
  }
  else if (dataReceive == 45)  //[-]-Key
  {
    encoderValue -= stepSize;
  }
  else if (dataReceive == 0) {
    encoderValue = stepper.currentPosition();
  }
  ...
```



Case Modeling

Challenges and Solution

For building our case we faced two crucial questions: How can we achieve a small overall size to maintain portability? Which technology can we use to create a case, which is both stable and lightweight? We quickly realized, that premade plastic cases for electronics would be too inflexible in terms of spatial efficiency, while the quality and stability was excellent. Our first approach for a self-made case was 3D-printing technology. First, we designed the parts with *CATIA*, a 3D modeling software by *Dassault Systemes* [2]. However, the printing took a very long time and the outcome was not satisfying. Even though we expected a maximum of freedom in terms of design possibilities, the printed parts of the case turned out to be very uneven and instable. Our third, and final option was the use of a laser cutter. We were able to reutilize parts of the 3D model, which we created for the 3D-printing approach. The 3mm wooden material turned out to be ideal for our case, as it offers the needed stability, while being lightweight and easy to process.

The finished case is composed of three compartments. The first one accommodates the electronic parts. A second one lies underneath and is designed exclusively for the batteries. The back of the case has two notches, which allow to open the backside easily when a battery exchange is due. In the front part of the case, there is space reserved for the stepper motor, which can be fixated with screws. The slots on the sides around the stepper motor are supposed to evacuate the heat produced by the motor when it is at work. Two holes drilled into the partition walls allow wires to connect the Arduino, stepper and battery compartments.

App Development

Challenges

For the development the of the remote controller app, we faced various challenges. As we defined the use of BLE as one of our requirements, only a part of the available Android devices is capable of executing our application, namely those with Android 4.3 or higher and the required hardware module built in. Currently (as of August 3, 2015), 62.1 percent of all Android devices meet the requirements software-sided, while the share of devices, which provide BLE hardware is unknown. [1] However, we did not consider this being a problem, as the advantages of BLE technology outweigh the disadvantage (compare *Bluetooth Low Energy* in *Our Follow Focus Approach*).

The other challenge was, how to implement the required functionalities. Here, it was important to keep the extensibility in mind. As the main goal is, to eventually embed the app and the hardware into to a larger system, it must be developed in a way, that it can be easily extended on the one hand, and that the logic can be transferred into another app on the other hand (compare *Our Vision*). One of the key challenges, was the implementation of the focus sequence recording in a way, that sequences are exactly reproducable.

Last, but not least, the User Interface (UI) should be created in a way, that makes the app easy to use, especially when it offers multiple different functionalities. As the design of a UI is not directly part of our requirements, we neglect this topic in this paper.

Our Solution

As our solution for the controller, we developed an Android App for smartphones. Through the provided UI, the camera operator is able to connect to the Follow Focus device via BLE. Once connected, the lens range of the attached camera can be calibrated in order to prevent the stepper motor from damaging the lens focus thread. The underlying logic for focus movement is relatively simple. The app manages two *byte*-values for both speed and direction of the movement. The speed value can be regulated via a slider element (*SeekBar* in Android SDK) with a range from 0 to 100. The direction can be either right (IN), left (OUT) or neutral (no movement), and can be regulated via two buttons. The default direction is neutral, and direction changes only, while one of the two buttons is pressed.
To implement the focus sequence recording function, we had to think of a way, how the user input could be saved including the temporal aspect. Therefore, the important information is not only *what* the user does, but also *when*. We basically came up with two approaches to this problem:

- Whenever a user input occurs, send the data and save the input type and value (e.g. speed 99), as well as the time delta from the beginning of the recording.
- Constantly check, save and send current speed and direction every *x* miliseconds.

As one of the key aspects of recording a sequence is reliable repeatability, we figured that the second option would offer the better results (in terms of result stability). Due to expected latency on both the master's side (i.e. the mobile android device) and the slave (the follow focus device), we expect the first option to lead to irregular playback result.
The sampling approach of the second option offers regular samples, which can be iterated through in the defined interval in order to exactly repeat the original input.

Listing 2. Focus Sequence Sampling

```
/*
* Timer Thread
* - executed every <EXECUTION_INTERVAL> ms
* - sends currentDirection and currentSpeed
*   via BLE to Arduino device
* */
public void startTimer (){
  final Handler handler = new Handler ();
  Timer timer = new Timer();
  timer.scheduleAtFixedRate(new TimerTask() {
    public void run() {
      handler.post(new Runnable() {
        public void run() {

// if not playing a recorded scene:
          if (!isPlayingScene) {

// send data via BLE

            if (isRecording) {
// if scene is currently recording,
// write values into FocusScene object
            }
          } else {
// if recorded scene is being
// played back, send recorded
// values as byte values.
// convert slide value (Integer)
// to byte values and send via BLE

// stop playing, when max frames reached:
            if (currentSceneFrame <
                selectedFocusScene
                .getSpeedValues().size()-1) {
              currentSceneFrame++;
            } else {
              currentSceneFrame = 0;
              isPlayingScene = false;
            }
          }
        }
      });
    }
  }, 0, EXECUTION_INTERVAL);
}
```

As can be seen in Listing 2, a timer thread in the controller app executes a function every *EXECUTION_INTERVAL* miliseconds. We defined this as an integer constant with the value of 50, which turned out to deliver the most stable results in our tests.

DISCUSSION & NEXT STEPS

CONCLUSION

REFERENCES

1. Android developers - platform versions.
2. Dassault systemes.

3. Lenzhound - kickstarter.
4. Proaim camera follow focus x1.
5. Andra motion focus, 2015.
6. Burgdorf, P. Take-over strategies and their effect on control and recording of automated camera tracking shots. Bachelor's Thesis, 2015.
7. Crabb, K. *The Movie Business: The Definitive Guide to the Legal and Financial Secrets of Getting Your Movie Made*. Simon and Schuster, 2005.
8. Mörwald, P. Development of a remote controlled camera slider. Bachelor's Thesis, 2014.
9. Soffer, A. Github - follow focus.
10. Stapleton, O. So you wanna work in movies? - camera department.