Synchronisation de deux caméras ToF

Les caméras ToF étant basées sur l'émission de lumière infrarouge, le fonctionnement en simultané de deux caméras nécessite une phase de synchronisation. Si cette synchronisation n'est pas faite, les infrarouges d'une caméra seraient interprétés par la deuxième et les mesures seraient faussées.

Pour résoudre ce problème, la synchronisation fait émettre les caméras tour à tour. Pour cela, le protocole PTP est utilisé (i.e. IEEE1588). La synchronisation des horloges internes des deux caméras et l'établissement des fréquences d'acquisition permettent aux caméras de ne pas se gêner.

Le fonctionnement de l'algorithme développé est le suivant :
- Connexion à la caméra
- Acquisition du mode de fonctionnement (Y a-t-il une ou deux caméras ?)
- S'il y a deux caméras, alors paramétrage et synchronisation.
- Acquisition des images

Nous documenterons ici seulement la partie de paramétrage et de synchronisation.

Les attributs :

```
bool WorkInPairs
```

Variable pour indiquer le fonctionnement simultané de deux caméras. True pour deux caméras et false pour une.

```
bool IsMaster
```

Lors de la synchronisation des horloges de deux caméras, l'une sera « Master » et l'autre « Slave ». Ce booléen indique le statut de la caméra : true pour Master et false pour Slave.

Les méthodes :

```
void BaslerTofCamera::SetupAndSync()
```

Méthode principale dans laquelle les réglages de la caméra sont effectués. La fonction ne prend aucun argument car elle travaille directement sur l'objet caméra. Elle ne revoie rien, elle modifie simplement certains paramètres.

```
        this->Set_Trigger_Mode(1);
        this->Set_Trigger_Source(3);
```

Ces deux lignes paramètrent la caméra de la façon suivante : Passe le trigger mode sur ON, la caméra sera donc en mode déclenché, elle aura sa fréquence d'acquisition déterminée par une horloge. Ensuite le protocole PTP est activé pour la synchronisation et la source de déclenchement est passée sur 3, ce qui équivaut au SyncTimer.

```
      this->Send_DataSet_Latch();
    CEnumerationPtr ptrGevIEEE1588StatusLatched =
m_Camera.GetParameter("GevIEEE1588StatusLatched");
    while (ptrGevIEEE1588StatusLatched->ToString() == "Listening")
    {
        this->Send_DataSet_Latch(); // Latch GevIEEE1588 status.
        cout << "." << std::flush;
        mSleep(1000);
    }
    Send_Info_Message("Status apres negociation : "+
      (string)ptrGevIEEE1588StatusLatched->ToString());
    if (ptrGevIEEE1588StatusLatched->ToString() == "Master")
    {
        this->Set_Is_Master(true);
    }
    else
    {
        this->Set_Is_Master(false);
    }
```

Ensuite, il y a la phase de négociation entre les deux caméras pour savoir laquelle est le maître et laquelle est l'esclave. Dans un premier temps nous récupérons le statut actuel de la caméra, puis tant que la négociation n'est pas terminée (i.e. le statut est « Listening ») on récupère le statut et on patiente. Une fois le nouveau statut déterminé, nous mettons à jour l'attribut IsMaster.

```
      const uint64_t tsOffsetMax = 10000;
    Send_Info_Message("En attente de la stabilisation de l'horloge sous "+
std::to_string(tsOffsetMax) +" ns ");
    if(!this->Get_Is_Master()){
        uint64_t tsOffset;
        do{
            tsOffset = GetMaxAbsGevIEE1588OffsetFromMasterInTimeWindow(1.0,
0.1);
            Send_Info_Message("Max offset of the camera = " +
std::to_string(tsOffset) + " ns ");
        } while(tsOffset >= tsOffsetMax);
    }
```

Finalement, il y a la phase de synchronisation. Dans cette partie, si la caméra est l'esclave, on attend simplement que son horloge se stabilise sous la valeur fixée par tsOffsetMax. Pour cela, on utilise la fonction qui renvoie la valeur absolue de l'écart entre les deux horloges.

```
int64_t BaslerTofCamera::GetMaxAbsGevIEE1588OffsetFromMasterInTimeWindow(double
timeToMeasureSec, double timeDeltaSec)
```

Lors de la synchronisation des horloges des caméras, il faut vérifier que l'horloge de la caméra esclave n'est pas décalée par rapport à l'horloge de la caméra maître. Cette méthode renvoie l'écart absolue entre l'horloge esclave et l'horloge maitre.

```
      this->Send_DataSet_Latch();
      int *ptrGevIEEE1588OffsetFromMaster;
      this->Get_Offset_From_Master(ptrGevIEEE1588OffsetFromMaster);
      StopWatch m_StopWatch;
      m_StopWatch.reset();
      int maxOffset = 0; // Maximum of offsets from master
      uint32_t n(0);  // Number of samples
      double currTime(0); // Current time
      do
      {
          currTime = m_StopWatch.get(false); // Update current time
          if (currTime >= n * timeDeltaSec)
          {
              // Time for next sample has elapsed.
              // Latch IEEE1588 data set to get offset from master.
              this->Send_DataSet_Latch();
              maxOffset = std::max(maxOffset,
              std::abs(*ptrGevIEEE1588OffsetFromMaster) ); // Maximum of offsets
from master.
              n++; // Increase number of samples.
          }
          mSleep(1);
      } while (currTime <= timeToMeasureSec);
      return maxOffset; // Return maximum of offsets from master for given time
interval.
```

```
void BaslerTofCamera::mSleep(int sleepMs)
```

Cette fonction permet de patienter le temps indiquer en argument en millisecondes.

```
#ifdef CIH_LINUX_BUILD
    usleep(sleepMs * 1000);   // usleep takes sleep time in us
#endif
#ifdef CIH_WIN_BUILD
    Sleep(sleepMs);           // Sleep expects time in ms
#endif
```

En fonction du système d'exploitation de la machine d'exécution, la valeur est en milliseconde ou en microseconde.

Des paramètres optimaux :

Dans le cadre de notre expérience qui était de faire fonctionner deux caméras en simultané à une fréquence de 20 Hz, nous avons déterminé les paramètres suivants :

HDR	0 (Standard)
ExposureAuto	0 (OFF)
SyncRate	20
ExposureTime	6 (ms)
IEEE2588	1 (ON)
TriggerMode	1 (ON)
TriggerSource	3 (SyncTimer)
TriggerDelay	Master → 0 (ms) Slave → 25 (ms)

Pour être sur d’avoir tout ces paramètres bien en place, une fois la fonction SetupAndSync() terminée, nous avons réalisé une série de Set/Get pour envoyer les paramètres à la caméra et vérifier qu’ils soient bien en place.