

Projet Programmation 2018

rev 1

Rendu

Date: xx juin 2018 à 23h55

Contenu de l'archive .zip

Rendu de vos fichiers dans une archive .zip aux noms des membres du groupe, avec:

- Brève documentation au format pdf
- Le(s) fichier(s) source(s) c++
- Le programme/exécutable (Pix2Pos) c++ compilé pour linux, mac ou windows
- Le(s) VI(s) LabVIEW
- Le(s) script(s) Matlab
- Un jeu de fichiers (*pixmap.bin*, *pos.txt*, *ComputeScore.m*, *AH_score.pdf*) générés par vos différents programmes

Contrôlez que **TOUS**** les fichiers se trouvent dans l'archive avant de la soumettre.**

Soumettre l'archive et vérifier ce que vous avez uploadé!



Salzmann.zip

Fichier
à soumettre

Rendu sur Moodle

- Déposer l'archive .zip via la page moodle du cours
- Nombre illimités de soumissions, uniquement la dernière est retenue

Soumission projet 2018



Soumission du projet 2018 sous la forme d'une archive .zip contenant l'entier de vos fichiers.




La taille maximum est de 50 MB.

Nommez votre archive avec les noms des membres du groupe.

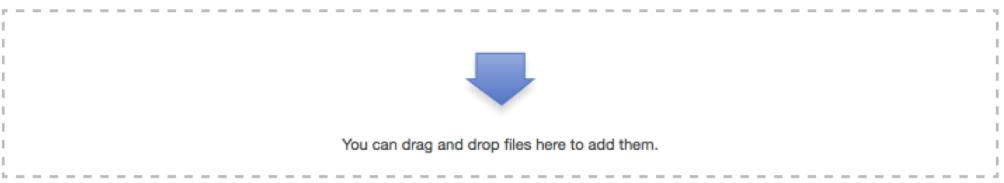
File submissions

Maximum size for new files: 50MB, maximum attachments: 1





► Files




You can drag and drop files here to add them.

Save changes

Cancel

Procédure d'évaluation

(A tester dans votre environnement avant la soumission!)

- Décompression de l'archive .zip
- Ouverture du fichier LabVIEW *AirHockey.vi*
- Choix des paramètres (couleur, etc.) dans l'interface LabVIEW
- Exécution du VI en cliquant sur la flèche 

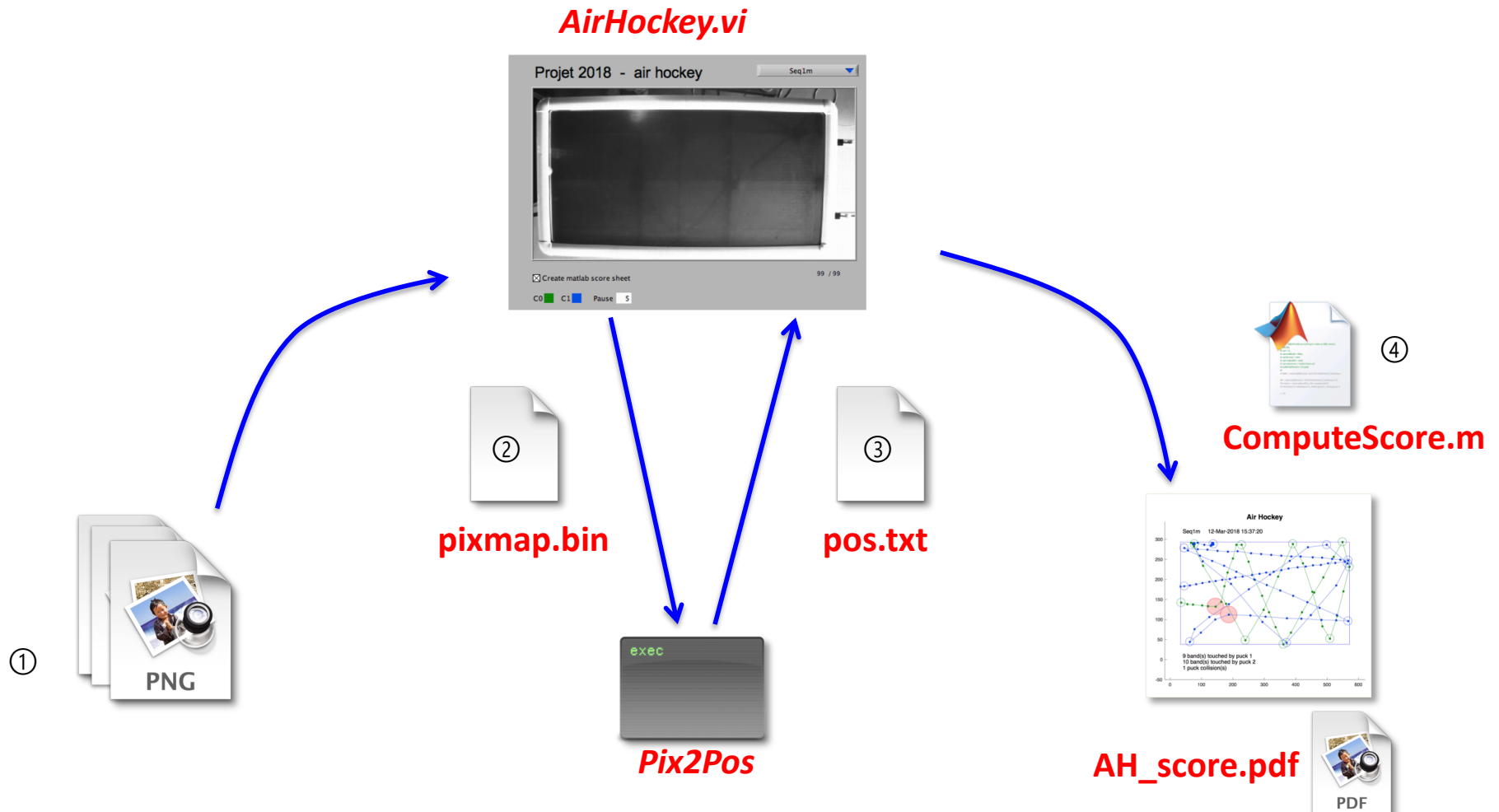
Après exécution du VI, est-ce que Matlab a généré correctement le fichier pdf contenant le graphique?

oui => vous êtes assuré/e d'avoir au minimum la moyenne pour la partie projet (*si vous n'avez pas triché avec votre code!*)

non => analyse de votre code et des documents fournis

Procédure d'évaluation, cont'

Respectez impérativement les **noms** des fichiers



Procédure d'évaluation, cont'

C++

- Votre code c++ est analysé en premier via le VI LabVIEW fourni et ensuite de manière indépendante en créant automatiquement des fichiers pixmap.txt et en observant les fichiers générés *pos.txt* ainsi que les messages éventuels dans *cerr* et *cout*
- **Le comportement de votre programme c++ est comparé avec les spécifications fournies durant le cours ainsi qu'avec votre documentation!**
- **Attention aux chemins relatifs, noms de fichiers, etc.**
- Analyse du(des) fichier(s) source C++

LabVIEW

- Test du bon déroulement du programme LabVIEW, test avec plusieurs jeux de fichiers et paramètres (valides et invalide), affichage du résultat dans matlab, **gestion des erreurs**, etc.
- Test avec exécutable c++ manquant
- Analyse de la gestion des erreurs en cas de problème avec la partie C++ ou LabVIEW (message(s) d'erreur et/ou autres mécanismes)
- **Comment gérez vous le cas d'une image invalide ? Décrire dans la doc.**
- Analyse du(des) fichier(s) source LabVIEW

Matlab

- Génération correcte du fichier *.pdf*
- Analyse du(des) fichier(s) source matlab

Test de validité des paramètres d'entrées

Votre code c++ doit pouvoir traiter les erreurs spécifiées dans le document «Projet2018.addendum.1.pdf » et générer un message d'erreur correspondant dans cout, cerr.
En fonction de la manière dont votre code est écrit, certaines erreurs décrites ci-dessus ne peuvent pas être détectées, dans ce cas vous devez l'indiquer dans le fichier de documentation.

Vous devez décrire **succinctement** dans vos spécifications comment vous gérez les erreurs ci-dessus. En particulier ce que vous faites en cas d'erreur, par ex. arrêt du programme ou utilisation de valeur par défaut, etc.

Spécifications

- Document PDF de 1 (**max. 5**) pages, pas besoin d'un roman, mise en page simple et bien organisée. Il doit permettre de comprendre votre démarche et la mise en œuvre de celle-ci.
- Contenu:
 - **Nom des auteurs**
 - La plateforme (linux, OSX, Win, ...) et le compilateur employés (xcode, geany, etc.)
 - Nom des fichiers et leur fonction
 - ex.
 - monVi.vi*: VI principal à lancer en premier
 - Le flow de données entre les applications et à l'intérieur des applications, i.e. les fichiers créés, lus et échangés entre les applications, + cout & cerr
 - ex. (éventuellement avec un schéma)
 - monVi.vi* lance *Pix2Pos*
 - Pix2Pos* lit le fichier *Pixmap.bin* et génère le fichier *pos.txt* ,
 - En cas d'erreur, *monPrg* **C/C++** fait... (détails svp)
 - En cas d'erreur, *monPrg* **LabVIEW** fait... (détails svp)

Spécifications - 2

- Contenu (suite):

- Courte doc. utilisateur

ex.

Entrez les données suivantes dans *monVi.vi* avant de le lancer: Choix des paramètres, ... puis faire

...

En cas d'erreur le message xy est affiché dans ...

- Autres informations utiles

ex. gestion des erreurs, ...

Si vous vous êtes inspiré de codes trouvés sur internet, mettez les références. Idem, si vous avez travaillé avec un autre groupe.

Précisions

- Les tests se feront sur des machines virtuelles linux, Windows ou sur mac
- Les spécifications et commentaires peuvent être en français ou en anglais
- Mettez des commentaires dans le code c++ , LabVIEW et matlab
- Relisez les spécifications, est-ce que vos programmes les suivent, ex. noms de fichiers, format, etc.
- **Ne pas mettre de chemin absolu dans vos codes c++/LabVIEW/matlab!**
-> pénalité si c'est le cas!
- **Mettez *TOUS* vos fichiers dans l'archive zip! Au besoin testez que votre archive fonctionne sur une autre machine**