Introduction

Bien que cela puisse paraître étrange, l'Art (la peinture, le dessin, l'infographie, la sculpture...) et les mathématiques sont deux domaines extrêmement liés. Les mathématiques sont certes une science exacte et l'Art relève davantage du domaine du sensible mais nous pouvons cependant remarquer que ce dernier utilise beaucoup les mathématiques.

La création d'une image par exemple répond à des critères simples de mathématiques. Le dessin ne peut être qu'un assemblage de données géométriques qui forment un contenu artistique : le pavage, apparu dans l'Antiquité, répond aux règles simples de la symétrie que ce soit dans l'utilisation de formes ou dans le déplacement par rotation ou translation. La peinture obéit également à des règles mathématiques : la création d'un tableau est pensée selon une répartition logique de l'espace définissant chaque partie de celui ci afin de former un contenu cohérent, tandis que la perspective utilise également la pensée mathématique.

L'exemple le plus concret reste tout de même le nombre d'or utilisé dans la Rome antique chez les architectes, les peintres, les sculpteurs et les dessinateurs qui cherchaient à allier imperfections et esthétisme dans une œuvre. Marcus Vitruvius Pollio, dit Vitruve définit le nombre d'or comme un rapport de proportion particulièrement esthétique qui consiste à établir "le rapport entre le tout et la plus grande partie de ce tout équivaut à celui qui existe entre la plus grande et la plus petite de ces parties" : cette idée sera retrouvée par la suite dans toutes les formes de la nature comme par exemple dans les spirale formée à l'intérieur des coquillages. Une des plus grandes figures dans le domaine de l'art et des mathématiques qui utilisa ce nombre d'or est Léonard de Vinci (1452-1519) : c'est à lui que nous devons la notion de perspective, ayant servi à un grand nombre d'artistes par le passé et aujourd'hui encore. Un de ses dessins les plus célèbres, tirant parti de nombreuses données mathématiques, est "l'homme de Vitruve" qui définit les proportions parfaites d'un corps humain.

Nous pourrions aussi parler de l'architecture ancienne et moderne qui allie Art et mathématiques ainsi que de bien d'autres sujets : mais il nous semble que les exemples précédents illustrent clairement le lien évident entre l'Art et les mathématiques.

Toute l'originalité de notre projet tient dans l'idée d'allier les Mathématiques à une forme d'art pour laquelle elle apparaît comme infiniment moins évidente, à savoir le théâtre. Si, bien sûr, il n'est pas question de les intégrer dans le texte d'une pièce - quoique l'on peut imaginer des scènes mathématiques, comme dans *La leçon* d'Eugène Ionesco ou encore avec la lecture du poème de Victor Hugo "j'étais alors en proie à la mathématique"-, le lieu théâtral se prête bien à l'intégration d'un contenu scientifique. Les nouvelles technologies, en ouvrant des possibilités scénographiques, ont également permis aux mathématiques d'entrer en scène. Mais notre réflexion ne s'est pas arrêtée là puisque nous avons voulu lier les émotions de l'acteur à une expression numérique. Si le théâtre est un art vivant, notre idée permet alors de traduire le plus finement possible l'expression même de la vie, à savoir les émotions...

Problématique

La partie scientifique d'un projet plus large

Ce travail s'inscrit dans un projet plus large dans lequel notre classe est engagée depuis la rentrée 2019. Il s'agit d'un projet Erasmus+ qui consiste en la réécriture plurilingue du mythe de *Dom Juan* de Molière utilisant comme sources les différentes versions existant dans les pays avec lesquels nous travaillons, à savoir l'Italie et l'Espagne. La lecture et la réécriture se font donc en plusieurs langues et à plusieurs mains. La réflexion scénographique, qui va de pair avec la réécriture puisque la restitution de nos travaux se fera sur scène, mélange plusieurs disciplines: les lettres, les langues, l'histoire mais aussi les mathématiques et les sciences. En effet, nous avons choisi d'approfondir la manière d'intégrer les sciences (physiques, mathématiques et numériques) dans la production artistique. Nous avons donc décidé de prendre comme mission la création de la scénographie de notre pièce et sa réalisation, entièrement numérique.

Les arts de la scène et le numérique

Si la scénographie correspond aux arts et techniques de l'aménagement de la scène et de l'espace théâtral, par *scénographie numérique* nous entendons les décors numériques de spectacles, la mise en scène d'événements numériques et la scénographie virtuelle de dispositifs entièrement dématérialisés. Nous utilisons le terme numérique car aucune réalisation "classique" par la peinture ou la sculpture n'est envisagée. Le création est entièrement liée à des projections de lumière ou d'images et couleurs.

De plus, ces choix peuvent être réalisés en amont ou de façon instantanée sur la base de données et de grandeurs physiques mesurées directement sur scène.

Notre idée est donc de jouer sur scène avec, comme fond, un décor numérique qui interagirait avec les acteurs. Le logiciel informatique afficherait le décor de fond avec des algorithmes mathématiques. Il serait possible d'afficher n'importe quel décor à partir du logiciel.

Développement

Matériel utilisé

Arduino :

Afin de créer ces décors de scène artificiels et numériques nous avons besoin d'utiliser Arduino (Arduino, et son synonyme Genuino[2], est une marque qui couvre des cartes électroniques matériellement libres sur lesquelles se trouve un microcontrôleur).

Notre objectif est de programmer des algorithmes mathématiques pour afficher et modifier les décors de manière dynamique sur les écrans une fois sur scène. Le microcontrôleur s'activera par exemple si l'un des élèves est plus ou moins proche d'un capteur de distance fonctionnant avec des ultrasons : ainsi s'afficheront sur les écrans des couleurs ou bien des diagrammes, équations ou tout autre représentation au lieu d'un décor réel ou encore d'un diaporama classique que l'on avance manuellement.

L'avantage d'Arduino est sa capacité, une fois programmé comme on le souhaite, à être pratiquement autonome. En effet, le microcontrôleur nécessite le téléversement d'un code informatique servant à l'initialiser puis à définir son usage courant qui est répété sous forme de boucle infinie et qui permet d'assurer le traitement de données récupérées en continu par les capteurs. Seule l'alimentation du dispositif reste indispensable. En outre, ces cartes électroniques sont compatibles avec de nombreux capteurs différents et peuvent enregistrer des données digitales (seuil) comme analogiques (continu).

Logiciel Max/MSP

Max/MSP est un logiciel inventé et développé par l'IRCAM. Il est utilisé principalement dans le domaine de la recherche musicale et dans la création de performances, concerts en live et dans des travaux mélangeant mathématiques et composition musicale. Mais son utilisation s'étend également à la représentation théâtrale et permettra notamment, dans le cadre de notre projet, de commander les changements de décors numériques en recevant les données acquises grâce à la carte Arduino et en effectuant un traitement supplémentaire.

Sa fenêtre de travail est constituée d'un *patch* où la conception du programme se fait de façon très intuitive en liant entre eux différents blocs de

fonctions qui permettent d'avoir une image très claire de l'algorithme que l'on veut faire fonctionner.

Travail sur les mesures de distances

Dans cette partie nous présentons le montage utilisé pour réaliser les interactions entre les acteurs et la scène grâce un capteur de distance.

Après avoir présenté le schéma de montage, nous illustrons les deux programmes qui permettent de réaliser l'interface entre Arduino et la scène. Enfin, nous détaillons les mesures effectuées pour étudier la réponse du capteur en fonction de la distance et pouvoir ainsi optimiser l'affichage sur la scène.

Montage

Le schéma en Figure 1 montre le montage réalisé pour les études de la réponse du capteur de distance. Le détecteur de distance est branché à l'alimentation à 5 V et à la Terre par les fils respectivement rouge et noir. Le signal sortant (ECHO) est envoyé à travers le fil bleu à l'entrée PIN 3. Le fil vert correspond au Trigger et nous permet, en donnant une impulsion HIGH durant 10µs, de déclencher la mesure. Le détecteur de distance est un module de détection d'ultrasons HC-SR04, ses caractéristiques techniques sont illustrées en Figure 2[1] . Concernant le principe physique de détection, le capteur produit une onde sonore à un instant t_0 et, si elle trouve un obstacle, l'onde réfléchie est détectée par le capteur même au temps t_1 . Le capteur renvoie donc l'intervalle de temps $\Delta t = t_1 - t_0$.

[1] source https://www.gotronic.fr/pj2-hc-sr04-utilisation-avec-picaxe-1343.pdf

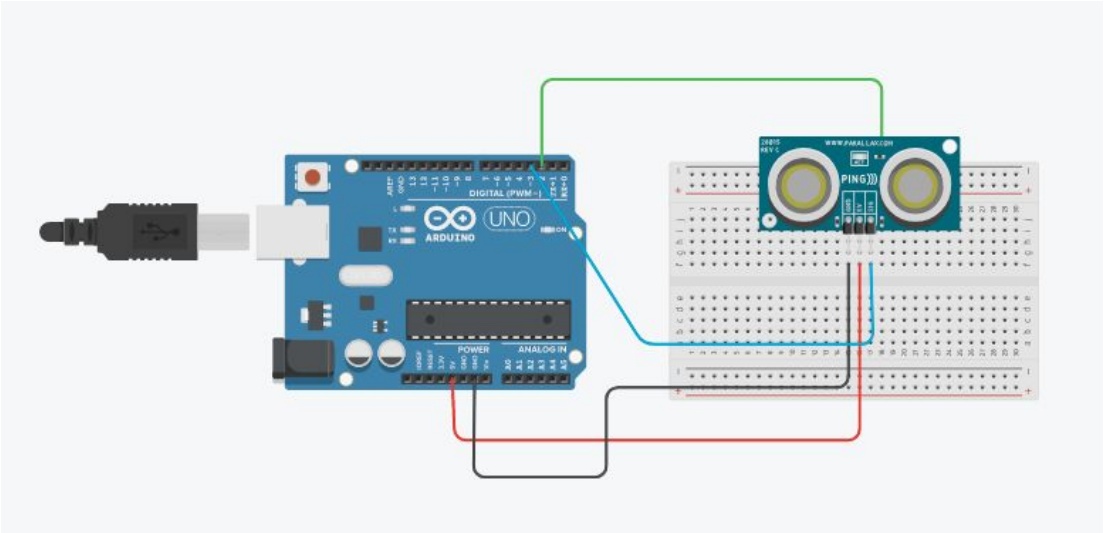


Figure 1 Schéma du montage du détecteur de distance

Caractéristiques

- Dimensions : 45 mm x 20 mm x 15 mm
- Plage de mesure : 2 cm à 400 cm
- Résolution de la mesure : 0.3 cm
- Angle de mesure efficace : 15 °
- Largeur d'impulsion sur l'entrée de déclenchement : 10 µs (Trigger Input Pulse width)

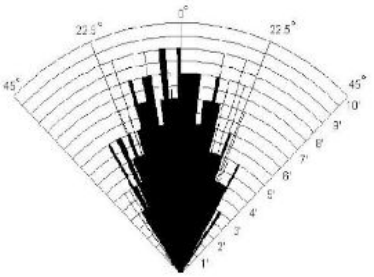


Figure 2 Caractéristiques techniques du module de détection d'ultrasons HC - SR04

L'annexe 1 présente le programme en langage Arduino que nous avons implémenté pour programmer la carte à l'acquisition des données.

La première partie "Constantes pour les broches" définit les entrées et les sorties, en particulier le TRIGGER et l'ECHO. Nous avons rajouté la sortie d'une LED (qui n'apparaît pas sur le montage) sur le pin 12 pour nous assurer du bon fonctionnement du capteur durant les phases du montage et, surtout, au moment de l'interface avec le logiciel Max (voir section suivante).

La programmation de la carte se fait en trois moments :

- Déclaration des constantes et initialisation: nous initialisons les différents PIN et nous définissons certaines constantes dont nous aurons besoin dans la suite du code, à savoir, le temps de mesure à 24 ms ainsi que la vitesse du son dans l'air à 0,340 mm/sec. Pour le choix du temps de mesure, nous avons considéré que la plage de mesure donnée dans les caractéristiques est de 400 cm. Nous avons donc considéré 800 cm pour la distance maximale d'un aller-retour de l'onde sonore envoyée par le détecteur. L'onde sonore se propageant à vitesse constante, nous avons calculé $\Delta t = \frac{\Delta x}{v}$ où Δt est le

temps de mesure cherché, Δx la distance de 800 cm et v la vitesse du son dans l'air. L'application numérique nous donne $\Delta t = \frac{800}{34000} = 0,00235\ s = 24\ ms$.

- La fonction *setup* fixe la fréquence de interface avec l'ordinateur aussi bien que les rôles d'entrée ou de sortie (input ou output) des broches.
  La fonction *void loop* lance la mesure par la commande
  long measure = pulseIn(ECHO_PIN, HIGH, temps_mesure);
  qui stocke l'information dans la variable *measure* : cette mesure est le temps d'aller-retour de l'onde sonore envoyée par le capteur.

- Enfin, la valeur de la distance est calculée par l'instruction
  float distance_mm = measure / 2.0 * vitesse;
  Nous avons également utilisé un bloc conditionnel pour vérifier notre montage et allumer une led si la distance entre l'obstacle et le capteur est inférieure à 500 mm. Après l'affichage des distances mesurées, un délai d'attente est envoyé au capteur pour ne pas avoir trop de données dans un intervalle de temps trop court.

Mesures de distance

Dans ce paragraphe une étude de la sensibilité du détecteur est détaillée. L'objectif des mesures réalisées était de caractériser le détecteur. Pour cela nous avons procédé à trois mesures différentes:

1. <u>Mesures de "petites" distances (1 - 50 cm).</u>
   Dans un premier temps, nous avons établi des distances de référence à l'aide d'un simple mètre. Nous avons mis un obstacle à chacune des distances établies et avons lancé le programme et donc la prise de mesure par Arduino (les résultats de ces mesures ont été transcrites dans le premier tableau où sont aussi précisés la différence entre la distance de référence et la distance mesurée, ainsi que le pourcentage de l'erreur relative entre les deux distances).

2. <u>Mesures de "grandes" distances (0,25 - 4,00 m).</u>
   De la même manière nous avons procédé aux mesures de distances sur un intervalle plus long (sans dépasser les 4 mètres) mais cette fois-ci en utilisant comme obstacle un camarade. Ce choix nous permet, notamment, d'avoir des résultats concrets quant à l'utilisation du capteur en situation réelle avec des acteurs sur scène (les résultats sont présentés de la même manière que précédemment dans le deuxième tableau).

3. <u>Mesures d'efficacité en fonction du matériau de l'obstacle.</u>
   Pour ces mesures-ci, nous avons fait l'inventaire des matériaux éventuellement présents sur scène et qui pourraient constituer le tissu des costumes des acteurs, mais aussi des parties de la scène. On trouvera parmi ces derniers : cotons, tissus, laine, différents vêtements : Jogging, Jean, Manteau, … Ici a contrario de ce que nous avons fait avant, nous avons cherché en fonction des matériaux, des vêtements exposés face au capteur à une distance de 1 mètre, la fréquence de 0 que l'on observait sur 50 lignes. En effet, l'observation d'un 0 correspond à un timeout sur le retour de l'onde, c'est-à-dire une réaction équivalente à une mesure pour un objet à plus de 4 mètres. Ceci nous permet de savoir quels matériaux absorbent le signal sonore, ce qui sera pris en compte lors de la mise en scène, car un matériau qui absorbe le signal sonore empêche toute détection de celui-ci, et entraîne donc "l'inutilité" du capteur. (Les résultats sont exposés dans le troisième tableau).

Mesure 1 Petit objet sur une table			
distance mesurée avec un mètre [cm]	distance mesurée avec Arduino [cm]	Différence	Pourcentage
1	2,4	1,4	140,00%
2	2,04	0,04	2,00%
3	2,7	0,3	10,00%
4	3,52	0,48	12,00%
5	4,8	0,2	4,00%
6	5,5	0,5	8,33%
7	6,92	0,08	1,14%
8	7,8	0,2	2,50%
9	9,11	0,11	1,22%
10	10,17	0,17	1,70%
15	15,5	0,5	3,33%
20	20,08	0,08	0,40%
25	24,7	0,3	1,20%
30	29,6	0,4	1,33%
35	34,3	0,7	2,00%

40	38,7	1,3	3,25%
45	44,2	0,8	1,78%
50	48,8	1,2	2,40%

Figure 3 Mesures de “petites” distances

A la vue des résultats, il est clair que le capteur présente des imprécisions de mesure allant de l’ordre du centième de centimètre (ex: 0,04) jusqu’à l’unité près (ex: 1,4). Ceci n’est évidemment pas négligeable, il faudra donc prendre en compte la marge d’erreur du capteur lors de la mise en scène, notamment pour le choix des distances entre les acteurs. Pour avoir une idée global de la précision du capteur nous avons cherché à comprendre quelle pouvait être son amplitude maximale. La colonne “pourcentage” nous permet de répondre à cette question. Partant d’une distance assez petite, le capteur présente un pourcentage de l’erreur très élevé, puis le pourcentage de l’erreur présente une certaine tendance à décroitre en phase avec l’augmentation de la distance et, par la suite, osciller entre 1% et environ 3%.

Avec les graphiques en Figure 4 et 5, construits à partir des données ci-dessus, nous observons que la proportionnalité entre les distances théoriques et celles mesurées est assez satisfaisante. Ainsi, nous pouvons en déduire, en trouvant l’équation de la droite qui mieux approche ces valeurs, la valeur hypothétique que le capteur est censé nous donner pour une quelconque distance. Cela nous permet de faire des prévisions pour des distances non étudiées, mais qui pourraient nous être utiles dans la pièce.

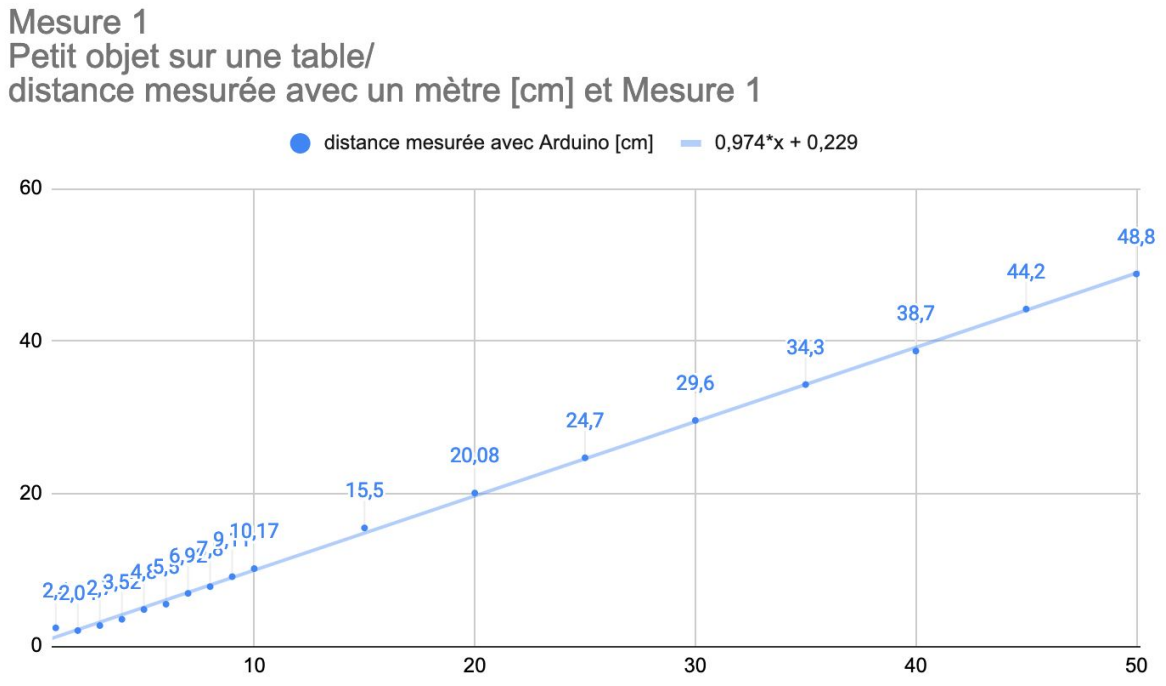


Figure 4 Nuage de points des mesures de “petites” distances

Certes, le fait d’utiliser un mètre pour les mesures de distances théoriques ne constitue pas une méthode très précise. Après une réflexion sur le rôle des erreurs de mesure que nous avons eues en classe, nous nous sommes dit que ce choix était cohérent avec l’usage que nous allions en faire sur scène où les distances entre les acteurs sont forcément “approximatives”.

Mesure 2 Un élève par rapport à Arduino habillé en tissu qui réfléchit ... (combin de sport?)			
distance mesurée avec un mètre [m]	distance mesurée avec Arduino [cm]	Différence	Pourcentage
0,25	0,245	0,005	2,00%
0,5	0,475	0,025	5,00%
0,75	0,733	0,017	2,27%
1	0,92	0,08	8,00%
1,25	1,22	0,03	2,40%
1,5	1,46	0,04	2,67%
1,75	1,7	0,05	2,86%
2	1,96	0,04	2,00%
2,25	2,2	0,05	2,22%
2,5	2,4	0,1	4,00%
2,75	2,71	0,04	1,45%
3	2,93	0,07	2,33%
3,25	3,18	0,07	2,15%
3,5	3,55	0,05	1,43%
3,75	3,71	0,04	1,07%
4	0	4	100,00%
4,25	0	4,25	100,00%
4,5	0	4,5	100,00%
4,75	0	4,75	100,00%

Figure 5 Mesures de “grandes” distances

Les mesures de distances longues semblent être plus précises que les courtes. Et cela, malgré le fait d'avoir utilisé comme obstacle un élève, beaucoup moins stable qu'un objet. Ceci nous conforte dans le fait de pouvoir utiliser ce système pour distinguer différentes phases dans le mouvement des acteurs sur scène et ainsi projeter des fonds différents avec une sensibilité de l'ordre des dizaines de centimètres.

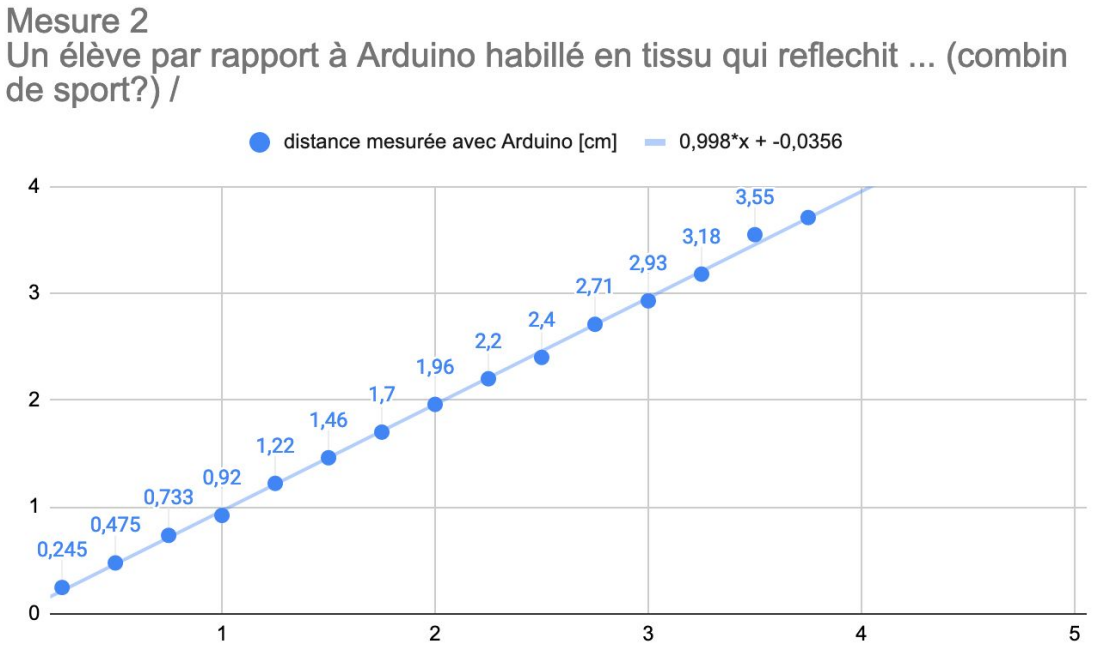


Figure 6 Nuage de points des mesures de "grandes" distances

Mesure 3 Différents matériaux	Fréquence des zéro observés (sur 50 mesures) Distance de mesure : 1 mètre
Coton (chemise?)	0/50
Laine (Pull)	36/50 (soit 72%)
autres (sweat, gilet, manteau, imperméable,... avec ceci ça marche très bien SI la distance n'est pas trop élevé ! (>4 mètre)	0/50

Figure 7 Etude des effets de différents matériaux

Lors de nos expériences nous avons effectivement observé que nombre de matériaux principaux composants de nos habits et donc de nos futurs costumes, n'absorbent pas le signal sonore émis par le capteur, ils le renvoient parfaitement. Cependant, il est clair que la laine est à éviter, car elle absorbe grandement le signal sonore émis (dans 72% des cas le signal a été absorbé), au risque que le capteur ne

puisse détecter la présence d'un obstacle et rende caduque toute l'utilisation des capteurs et du programme informatique.

Interface avec la scène

L'interaction avec la scène de la pièce de théâtre est réalisée grâce au logiciel Max/MSP[2] dont le lycée nous a acheté une licence pour un an.
L'idée du schéma de programmation est la suivante:

1. Le logiciel lit l'information sortant de la carte Arduino par le port USB auquel elle est reliée.
2. L'information (distance en cm) rentre dans un bloc conditionnel qui allume l'affichage d'une couleur selon la réalisation ou pas de la condition du bloc.
3. L'image est projetée en arrière plan sur la scène par un simple vidéoprojecteur.

L'annexe 2 montre la page de programmation que nous avons réalisée sur le logiciel Max.

La première partie contient l'interface de communication avec la carte Arduino, il est à remarquer l'utilisation de la même fréquence de communication que celle utilisée dans la phase de programmation de la carte.

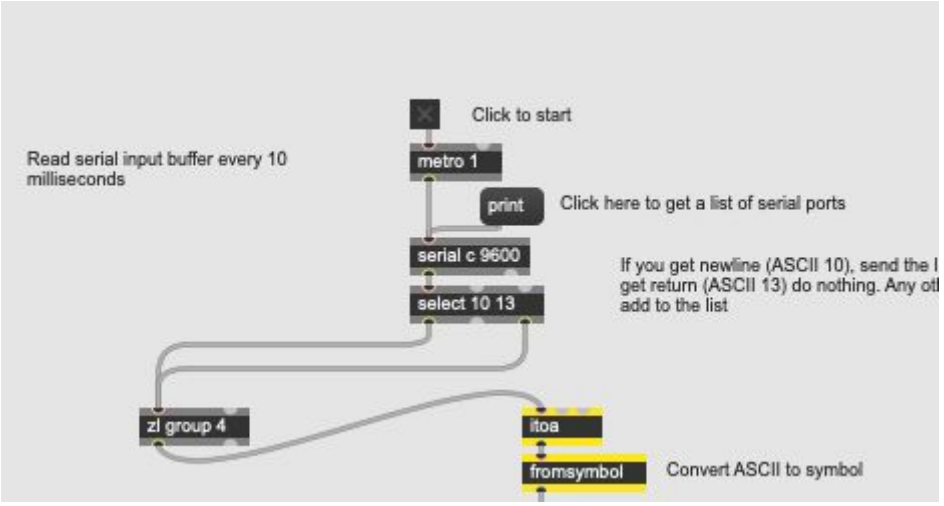


Figure 8 Mise en place de la communication entre Arduino et Max

La deuxième partie (Figure 8) est constituée du bloc conditionnel, la condition est remplie si la distance entre le détecteur et l'obstacle est inférieure à 300 cm.

[2] https://cycling74.com/products/max

Enfin, la sortie du bloc conditionnel est reliée à l'apparition d'une couleur. Dans la prise d'écran proposée, la couleur bleue correspond à une distance de plus de 300 cm tandi que la couleur rouge correspond à moins de 300 cm (voir Figure 9).

Le logiciel peut tester plusieurs conditions correspondant à plusieurs seuils de distance et ainsi permettre l'affichage de plusieurs couleurs différentes.

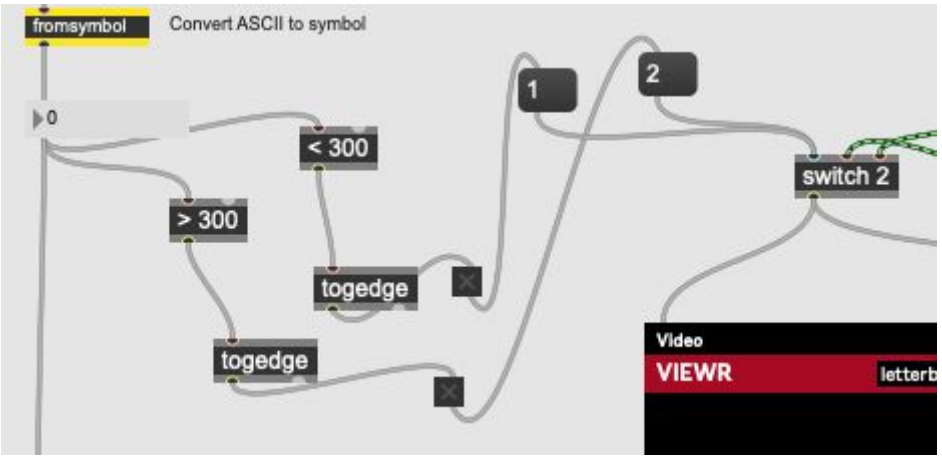


Figure 9 Bloc conditionnel dans la programmation du logiciel Max

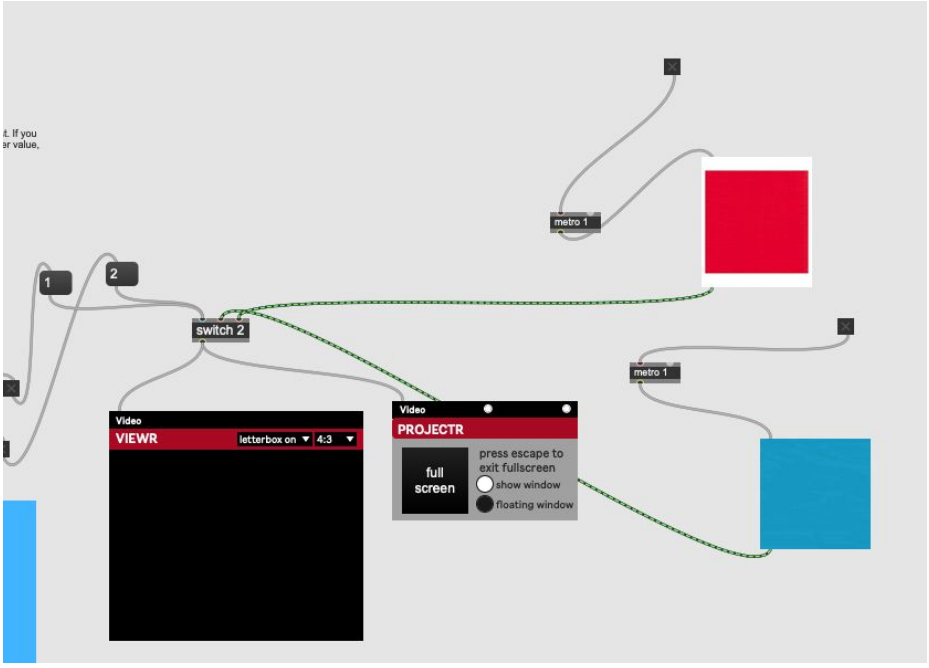


Figure 10 Choix de couleur en fonction de la sortie du bloc conditionnel

Intégration de la scénographie numérique à la pièce écrite

Les études réalisées pour la caractérisation du détecteur de distances nous ont permis d'imaginer son utilisation pour la scénographie de notre pièce de théâtre.

A titre d'exemple, nous avons choisi des scènes de *Dom Juan* qui se prêtent le mieux à l'utilisation de ce dispositif. L'objectif général est d'utiliser notre montage pour afficher sur le fond de la scène des décors qui puissent évoquer les émotions exprimées par le texte.

Les émotions sont la colère, l'amour et la peur.

Voici en détail les interactions que nous envisageons :

Pour la scène 3 de l'acte I, lorsque Don Juan est face à Elvire et qu'il la délaisse et demande à Sganarelle de parler à sa place, Elvire est en colère et s'approche de Don Juan ; nous avons donc imaginé un fond qui passe du orange au rouge pour montrer la montée de colère d'Elvire lorsqu'elle s'approche de Don Juan.

Pour la scène 2 de l'acte II, lorsque Don Juan rencontre la paysanne Charlotte et qu'il la charme, nous avons imaginé un fond de la couleur rouge avec des battements qui s'accélèrent au fur et à mesure qu'ils se rapprochent. Les battements pour faire penser au coeur et le rouge pour l'amour.

On passerait de 90 battements par minutes (ce qui correspond à une fréquence cardiaque moyenne à repos) à 120 puis 150, ce qui correspond à une fréquence cardiaque très élevée (comme pendant un entraînement)

Pour la scène 7 de l'acte IV, lorsque Sganarelle se retrouve face à la statue et qu'il est pris d'une grande peur, nous avons imaginé un fond bleu foncé qui s'éclaircit au fur et à mesure qu'il s'éloigne d'elle. Le choix du bleu étant une allusion à l'expression « peur bleue ».

Travail sur la détection de bruits

Dans cette partie nous présentons le montage réalisé pour l'utilisation d'un capteur de bruit en interaction avec la scénographie numérique.

Suivant la même structure de la partie sur l'étude des distances, nous présentons le schéma de montage, nous illustrons les deux programmes qui permettent de réaliser l'interface entre Arduino et la scène. L'étude de la réponse du détecteur n'est pas encore finalisée et nous choisissons d'en présenter seulement l'idée.

Montage

Le schéma en Figure 11 montre le montage réalisé pour les études de la réponse du capteur de bruit. Le détecteur de bruit est branché à l'alimentation à 5 V et à la Terre par les fils respectivement rouge et noir. Le signal sortant est envoyé à travers le fil bleu à l'entrée analogique A0. Le détecteur de bruit est un capteur sonore GT1146. Concernant le principe physique de détection, le capteur est doté d'un microphone alimenté par les 5V de la carte Arduino et donne un signal sortant binaire qui correspond à la détection ou pas d'un son d'intensité supérieure à un seuil donné réglable par une vis (potentiomètre).

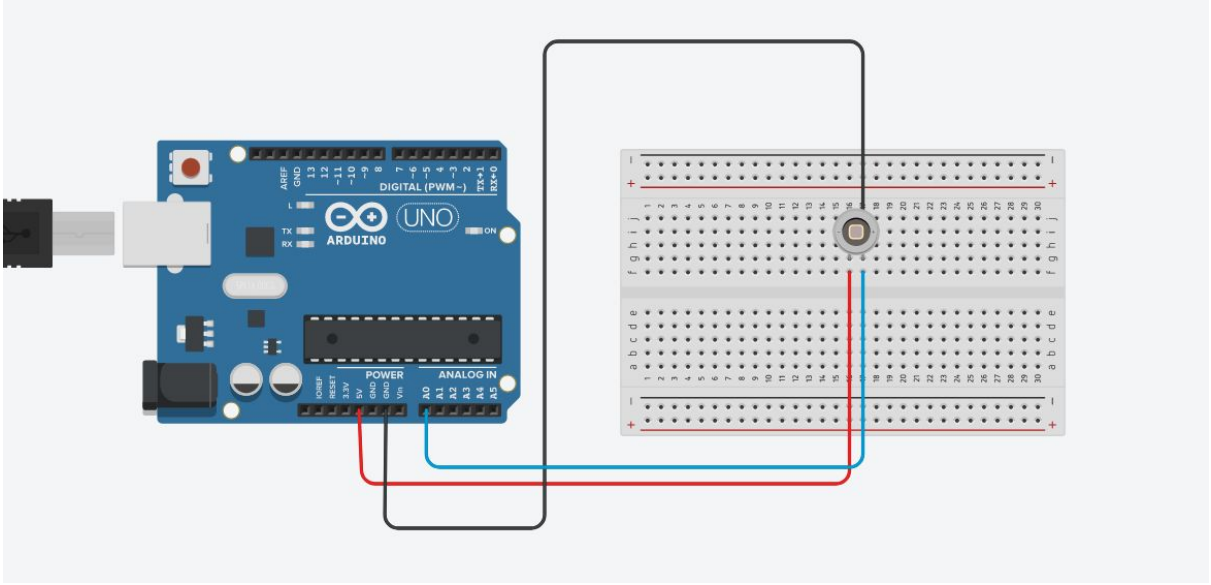


Figure 11 Schéma du montage du détecteur de bruit

L'annexe 3 présente le programme en langage Arduino que nous avons implémenté pour programmer la carte à l'acquisition des données.
Ce programme est beaucoup plus simple que celui que nous avons utilisé pour la détection des distances et consiste en l'initialisation de la broche A0, la déclaration

de la variable *mesure* et une fonction *loop* qui lit en continu la sortie analogique du microphone.

Interface avec la scène

L'interface avec la scène est réalisée grâce au programme Max en annexe 4.

L'idée de cette interface est de projeter une figure sur le fond de scène quand un bruit est détecté.

Si la partie de communication avec la carte Arduino est identique au programme utilisé pour l'exploitation des mesures de distance, la structure conditionnelle est plus complexe.

En effet, comme la mesure du niveau sonore se fait en continu, le logiciel Max reçoit un flux continu de données et les valeurs qui dépassent le seuil fixé par le capteur ne constituent qu'une petite partie de l'ensemble des données (imaginer la durée du bruit provoqué par un applaudissement par exemple). Ainsi, l'affichage d'une image conditionné par la détection ou pas du bruit serait très court voire invisible.

Pour pallier ce problème, nous avons intégré une fonction d'attente (le bloc *qmétro*) et une fonction de compte à rebours (le bloc *counter*) qui permettent, une fois la condition remplie, de laisser l'image choisie apparente pendant un temps que nous pouvons décider.

Le choix de l'image est réalisé suivant le même protocole que celui illustré dans la détection des distances.

Les résultats visuels sont montrés dans la vidéo qui accompagne ce rapport dans laquelle nous avons imaginé une possible application concrète de ce programme à une situation de scène qui interagit avec un bruit (dans ce cas précis la voix du père).

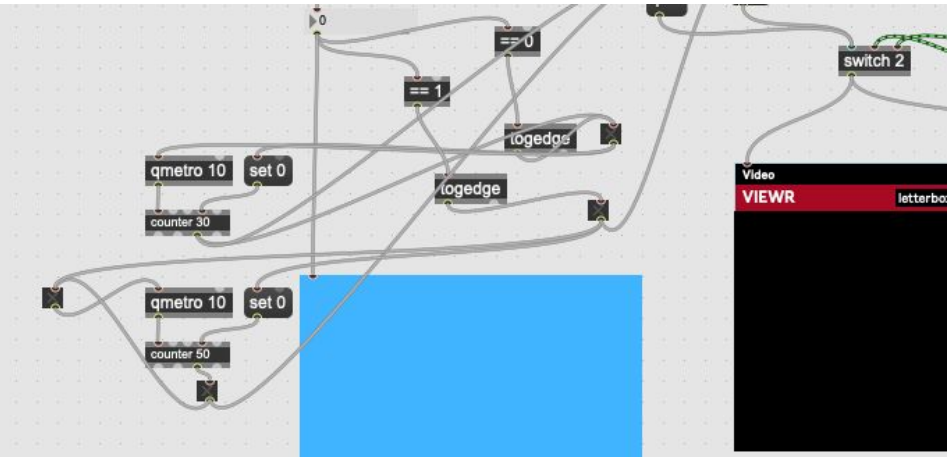


Figure 12 Blocs additionnels pour la gestion du temps d'apparition des images

Intégration de la scénographie numérique à la pièce écrite

A partir des scènes que nous avons écrites dans les deux premiers actes - le premier étant écrit par notre lycée et le deuxième par le lycée espagnol-, nous avons commencé à sélectionner des scènes qui pourraient donner lieu à des mesures de son. La lecture des travaux qui nous ont été envoyés nous a permis d'isoler ces deux premières répliques : les termes choisis pour entraîner une modification du motif projeté sur l'écran se trouvent dans les extraits ci-dessous.

Acte I

Sganarelle- *(Se levant d'un bond jetant les habits sur la tête de Godefroy)* **Diantre** ! En parlant du loup ! Le voici qui arrive !

Don Juan : Sganarelle, que penses tu de cette citrouille ?

Sganarelle (affolé) : **Malheureux**, que faites vous ! Ne saviez vous pas que les citrouilles sont hantées, le 31 octobre prochain la faucheuse viendra certainement à vous...

Bien accentués et prononcés avec force, les termes soulignés en caractère gras sont susceptibles de faire varier nos projections sur les écrans. Nous avons remarqué au cours de nos différentes expériences que l'effet était accentué par les mains posées en conque autour de la bouche : sans doute nous faudra-t-il rajouter cela dans les didascalies !

De la même manière, nous avons choisi dans les textes écrits par nos camarades espagnols deux répliques qui nous semblent susceptibles de produire des effets similaires. Cette fois Dom Juan ne crie pas mais il gifle son valet, comme on le lit souvent dans les didascalies des comédies qui mettent en scène des maîtres et les valets au XVIIe siècle. Le geste doit être simulé car en dehors du confort de l'acteur sur scène, le bruit doit être fort : la "gifle" doit produire un claquement sec afin que le son émis soit capté et peut-être sera-t-il nécessaire d'utiliser un trucage pour obtenir un volume suffisant afin que le motif projeté à l'écran du fond de la scène se modifie. La répétition de ce geste au cours des

scènes écrites par nos camarades espagnols permettrait ainsi d'obtenir un effet comique par répétition du motif choisi.

Acte II

Scène 3

PERICO: No creo estar confuso con lo que acabo de ver.
*(Don Juan le da un **bofetón** a Perico).*
PERICO: No se atreva a volver a pegarme.
*(Don Juan le da un segundo **bofetón**)*.

Scène 6

Don Juan: (le da una **bofetada**) ¿Cómo osas hablar así a tu señor? Muestra más respeto.
Ciutti: Disculpe mi osadía, no volverá a ocurrir.

Conclusions et perspectives

Dans ce rapport nous avons illustré l'état d'avancement de notre projet de réalisation d'une scénographie entièrement numérique pour la mise en scène d'une réécriture du mythe de Dom Juan. Cette pièce sera le résultat d'une écriture plurilingue que nous réalisons avec nos partenaires espagnols et italiens dans le cadre d'un projet Erasmus+.

Le travail scientifique est basé sur la mesure et l'exploitation de deux paramètres du spectacle : la distance entre les acteurs et le volume sonore.

Ces paramètres sont mesurés par deux capteurs intégrés à des cartes Arduino et leurs valeurs analysées et exploitées par le logiciel Max qui nous permet un affichage d'images aussi bien que de vidéos sur le fond de la scène.

Nous avons montré les montages et les programmes réalisés à cet effet. Pour les mesures de distance, nous avons produit une étude de linéarité de la réponse du capteur pour des "petites" et "grandes" distances et une analyse qualitative de la stabilité de la réponse en fonction du matériau du costume de scène qui joue le rôle d'obstacle pour le détecteur.

Les résultats montrent un comportement assez linéaire dans l'intervalle de distance qui va de 10 cm jusqu'à 3,75 m. Ceci nous permet, notamment, d'imaginer des situations variées d'interaction entre la scène et le décor comme, par exemple,

des couleurs qui illuminent le fond de scène en fonction de la distance entre deux acteurs avec des crans de 20 cm. Cela permet de créer un effet de nuance qui peut être associé par exemple à des sentiments de jalousie ou d'amour ainsi que de rage, sentiments présents tout au long de la pièce et très liés au personnage de Dom Juan.

Nous avons étudié les effets d'instabilité potentiellement dus aux matériaux du costume de celui qui s'approche du détecteur. En effet, un seul tissu en laine nous a donné un résultat très peu satisfaisant avec un nombre important de 0 cm détectés. Ceci peut être mis en lien avec l'absorption des ondes sonores par la laine aussi bien que par la structure du tissu qui ressemble à un réseau dont les trous fonctionnent comme des "trappes" pour l'onde sonore.

Enfin, un montage pour l'utilisation d'un détecteur de sons a été réalisé et expliqué et les enjeux de la programmation pour faire l'interface avec la scène ont été détaillés. Ce montage a été utilisé, notamment, pour la réalisation d'une vidéo d'essai qui nous a permis de nous apercevoir des beaux résultats que ce projet peut donner.

La première représentation de la pièce de théâtre se jouera le 27 mai 2020 à 19h au théâtre La Merise de la ville de Trappes.

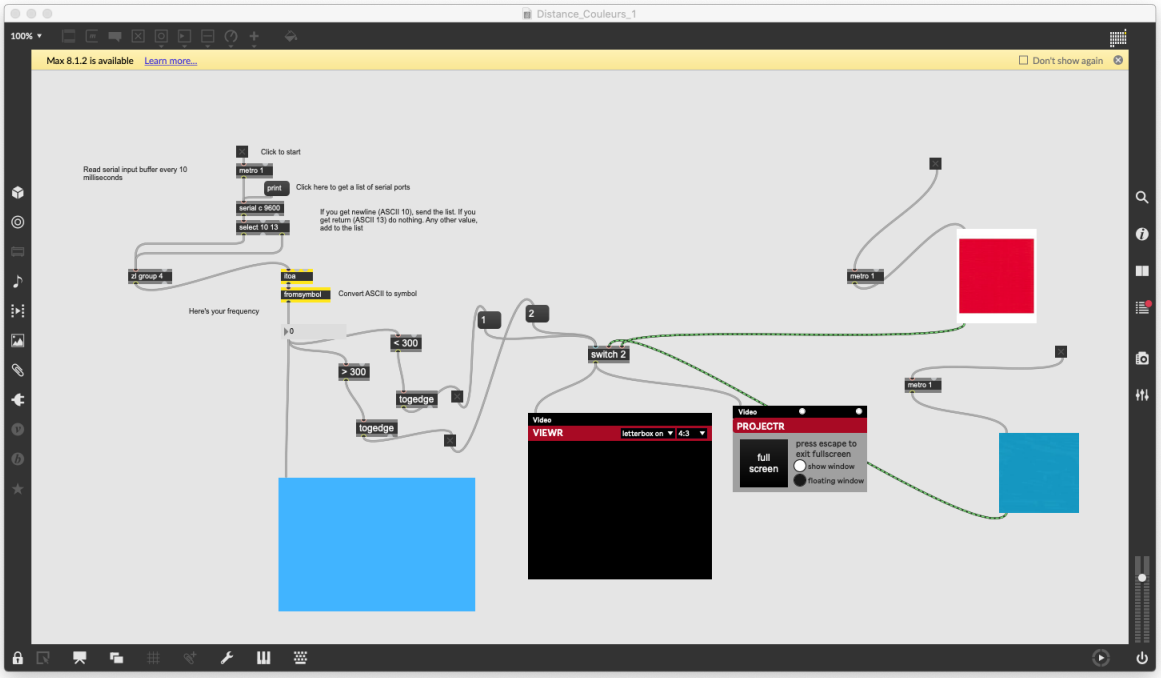
Annexe 1

Programmation de la carte Arduino pour la détection de distances

```
const byte TRIGGER_PIN = 2; // Broche TRIGGER
const byte ECHO_PIN = 3;    // Broche ECHO
const byte LED = 12;

const unsigned long temps_mesure = 24000UL; // 25ms = ~8m à 340m/s

const float vitesse = 0.340;


void setup() {

 Serial.begin(9600);


 pinMode(TRIGGER_PIN, OUTPUT);
 digitalWrite(TRIGGER_PIN, LOW);
 digitalWrite(LED, LOW);
 pinMode(ECHO_PIN, INPUT);
}

void loop() {

 digitalWrite(TRIGGER_PIN, HIGH);
 delayMicroseconds(10);
 digitalWrite(TRIGGER_PIN, LOW);
 long measure = pulseIn(ECHO_PIN, HIGH, temps_mesure);

 float distance_mm = measure / 2.0 * vitesse;
 if (distance_mm<500) digitalWrite(LED, HIGH);
 else digitalWrite(LED, LOW);

 Serial.print(F("Distance: "));
 Serial.println((int)distance_mm);
 Serial.print(F("mm ("));
  delay(1000);
}
```

Annexe 2

Fichier du logiciel Max pour l'interaction entre les mesures de distance et la scène



Annexe 3

Programmation de la carte Arduino pour la détection de sons

```
const byte Amplitude = A0; // Sortie du microphe
int measure1 = 0;

void setup() {

 Serial.begin(9600);

}

void loop() {


 measure1 = analogRead(Amplitude);

 Serial.println(measure1, DEC);

 delay(100);
}
```

Annexe 4

Fichier du logiciel Max pour l'interaction entre la détection de bruit et la scène