11: procédure de dessin

introduction

Dans ce didacticiel, nous allons utiliser plusieurs nouveaux objets pour réaliser un dessin procédural avec l'objet *lcd*. L'objet *counter* fournira un traitement pas à pas énuméré, l'objet *line* nous permettra de créer une sortie lissée et l'objet *scale* nous permettra de modifier la plage de sortie. Enfin, nous allons présenter quelques objets mathématiques qui nous aideront à créer des formes courantes nécessitant une trigonométrie simple.

Nous utilisons souvent des constructions programmatiques et procédurales pour effectuer des tâches dans nos programmes; celles-ci peuvent présenter de nouvelles opportunité pour l'affichage graphique, mais peuvent également être utilisées chaque fois que nous devons compter sur l'aide d'un ensemble de nombres, créer une trajectoire numérique ou des valeurs d'échelle. Qu'il soit utilisé pour la visualisation audio, le retour d'information des contrôleurs ou simplement pour un art intéressant, le code procédural peut servir à la fois d'outil utile et d'inspiration.

Pour ouvrir le patch du didacticiel, cliquez sur le bouton vert **Ouvrir didacticiel** dans le coin supérieur droit de la fenêtre de documentation.

Utiliser *counter* pour dessiner

Jetez un coup d'œil au patch du tutoriel. Ce patch comporte quelques zones qui nous aident à comprendre certains nouveaux objets et un autre grand patch de dessin. En haut à gauche se trouve un petit patch qui illustre l'objet *counter*. Lorsque nous cliquons sur le *bouton* de gauche, l'objet produit une sortie numérique qui s'incrémente de **1** pour chaque message bang reçu. Lorsque la sortie atteint **9**, le **bang** suivant réinitialisera la sortie à **0**. Les arguments que nous avons fournis au *counter* ici fournissent les nombres **minimum** et **maximum** pour la sortie. Lorsque l'objet atteint son maximum, il retourne au nombre minimal. Cela nous permet d'utiliser des *counters* pour créer des boucles de valeurs.

L'objet *counter* est quelque peu inhabituel en ce sens qu'il peut accepter un nombre variable d'arguments:

> Si vous avez **un** argument, il définit la valeur **maximale** du *counter*.
>
> Si vous avez **deux** arguments, ils définissent les valeurs **minimale** et **maximale** du *counter*.
>
> Si vous avez **trois** arguments, ils définissent la direction, les valeurs **minimum** et **maximum** du *counter*.

Notre patch de test pour *counter* comprend un *bouton* supplémentaire (coloré en rouge) et deux boîtes de nombres pour définir les valeurs de l'objet. C'est une excellente occasion d'utiliser l'**assistance** sur les entrées pour savoir ce qu'elles font. Si nous **déverrouillons** le patch et que nous survolons la troisième entrée (dotée d'un *bouton* et d'une boîte de *nombre* connectés), l'assistance dit «Réinitialise le *counter* sur le nombre à la prochaine horloge». Cela nous permet de modifier la valeur actuelle du *counter* pendant son exécution.

Si nous modifions la boîte de *nombre* à **5** puis appuyions sur le premier *bouton*, nous voyons que la sortie est le nombre **5** - quelle que soit la valeur précédente. C'est ainsi que nous définissons l'état de *counter* à la volée. En cliquant sur le deuxième *bouton* (rouge) (qui envoie le message **bang**), la valeur sera réinitialisée au minimum: **0** dans ce cas.

La boîte de *nombre* à droite nous permet de modifier la valeur maximale à la volée. Cela serait utile pour les situations de dessin où vous devez modifier la taille de la zone de dessin ou lorsque vous séquencez un effet et que vous devez modifier le nombre d'étapes. Lorsque nous modifions cette valeur et que nous appuyons plusieurs fois sur le *bouton*, nous constatons que la nouvelle valeur maximale est respectée.

L'objet patcheur intitulé **exemple 1** est celui où nous utilisons l'objet *counter* pour dessiner de manière procédurale dans l'objet *lcd*. Lorsque nous démarrons le *metro* (à l'aide de l'objet *toggle* connecté), nous voyons que le patch commence à dessiner de petits cercles dans l'écran *lcd*, la couleur changeant progressivement au fur et à mesure de sa progression. Lorsque le dessin atteint le bas de l'écran *lcd*, il recommence à dessiner en haut.

• Double-cliquez sur l'objet patcheur nommé **exemple 1** pour l'ouvrir.

Quatre objets *counter* différents sont utilisés dans ce patch; un pour définir l'emplacement et la taille du dessin, et trois autres pour modifier la couleur. Comprendre le *counter* le plus à gauche est le plus important: il compte de **0** à **767**, soit **768** étapes au total. L'emplacement X / Y du dessin est calculé avec les objets *%* (modulo) et */* (division), ce qui décompose notre objet *lcd* en une grille de **3 2** x **32**. Si nous regardons la sortie de *counter* et des objets *%* et */*, nous pouvons voir que le système fonctionne en créant deux boucles allant de **0** à **31**, l'une passant rapidement (à partir de l'objet *%*) et l'autre s'incrémentant à chaque fois que la première boucle est répétée (l'objet */*). Les valeurs sont multipliées par **10** (pour obtenir l'emplacement du pixel dans le coin supérieur gauche de la forme), puis on leur ajoute **10** (pour obtenir l'emplacement du pixel dans l'angle inférieur droit). Ceci établit une grille de cercles remplissant le *lcd* et que notre programme doit dessiner.

Les trois autres objets *counter* parcourent trois plages de nombres différentes pour nous donner une palette de couleurs en constante évolution. Ces objets *counter* utilisent l'initialisation à trois arguments, où le **2** initial force *counter* à se déplacer vers le haut jsqu'au maximum, puis vers le bas jusqu'au minimum. C'est ce qui fait que le dessin change lentement de couleur au fur et à mesure qu'il se dessine. En cliquant sur le *bouton* "réinitialiser", la simulation reprendra depuis le début de la boucle.

Utiliser *scale* pour dessiner

Si nous revenons à la gauche du patcheur, le deuxième petit exemple montre l'utilisation de l'objet *scale*. Cet objet vous permet de définir une plage de valeurs d'entrée et une plage de valeurs de sortie. l'objet mettra ensuite correctement **à l'échelle** les valeurs afin qu'elles soient représentées dans la nouvelle plage de nombres. Dans notre exemple, les valeurs d'entrée comprises entre **0** et **100** correspondent à la plage de **-1,0** à **1,0** - ceci est déterminé par les arguments fournis à l'objet. Si vous modifiez l'entrée de la boîte de *nombres*, vous constaterez qu'une valeur de **0** génère une sortie de **-1,0**, une valeur de **50** nous donnera une valeur de **0.0** et une valeur de **100** nous donnera un résultat de **1.0.** Ces nombres sont représentés de manière linéaire sur toute la plage fournie.

Si vous choisissez de dépasser la plage d'entrée, vous constaterez que la sortie continue à utiliser l'effet de plage calculé, vous permettant ainsi de dépasser la plage attendue sans échec. Par conséquent, une valeur d'entrée de **150** se traduira par une sortie de **2,0**, tandis qu'une valeur de **-100** vous donnera une valeur de **-3**.

L'objet *patcheur* nommé **exemple 2** utilise *scale* pour dessiner une forme basée sur des calculs de sinusoïdes. Un objet *counter* génère une sortie allant de **0** à la valeur maximale (définie à l'aide des deux boîtes de *nombres* situées à droite de l'objet *metro*). Celui-ci est mis à l'échelle de **0,0** à **6,283185** (2 * pi), puis envoyé à un objet *sin*, qui calcule le sinus de la valeur en entrée. Cela

produit une onde sinusoïdale dont la plage varie de **-1,0** à **1,0**. Cette valeur est envoyée à un autre objet *scale* qui fait correspondre cette plage à la plage d'entiers **0** à **319** - la plage de taille de l'objet **lcd**. Ceci est utilisé pour fournir les emplacements en X de la ligne. Un ensemble similaire de fonctions est utilisé pour définir l'emplacement en Y. Ces nombres sont ensuite insérés dans une liste à l'aide de *pack*, attachés à un message **lineto** (à l'aide de *prepend*), puis envoyés à l'écran *lcd*.

Effacez l'écran *lcd* à l'aide de la boîte de *message* **clear**. Activez le *metro* à l'aide du *toggle* et observez le résultat. Le résultat est un dessin à base de sinus sur toute la surface de l'écran *lcd*. Vous pouvez définir les valeurs maximales de la procédure en modifiant les boîtes de *nombres* attachées aux objets *counter*. Notez que les boîtes de *nombres* changent non seulement les paramètres des objets *counter*, mais aussi la plage d'entrée maximale attendue par les objets *scale* situés immédiatement en dessous (la troisième entrée de l'objet *scale* détermine la plage d'entrée maximale attendue).

Les deux objets *counter*, lorsqu'ils sont réglés sur des points maximum différents , créent différentes ondes sinusoïdales imbriquées qui se propagent sur l'écran *lcd*. Ces courbes sont appelées modèles de **lissajou** et simulent ce qui se produit lorsque des oscillateurs déphasés sont introduits dans les entrées X et Y d'un oscilloscope. C'est un excellent exemple d'utilisation de fonctions mathématiques simples pour créer des dessins complexes.

Utiliser *line* pour dessiner

En revenant en haut à gauche du patch, le troisième exemple nous donne l'occasion de voir l'objet *line* en action. L'objet *line* prend une paire de valeurs entrantes (configurées en tant que paire valeur / temps) et se déplace lentement d'une valeur à la suivante. Si une seule valeur est reçue, il saute immédiatement à cette valeur. Cependant, lorsqu'une deuxième valeur est fournie, elle est traitée comme le nombre de millisecondes utilisé pour passer à cette nouvelle valeur.

En cliquant sur la première boîte de *message* (intitulée **100 1000**), *line* passera de la valeur de départ de **0** à la valeur cible (**100**) sur **1 000** millisecondes. En cliquant à nouveau dessus, vous ne verrez aucun changement, car la valeur actuelle (qui reste à **100**) est identique à la valeur cible (elle n'a nulle part où aller). L'objet *line* est « sensible à l'état », ce qui signifie qu'il comprend sa valeur actuelle et ajuste son action en fonction de l'état de cette valeur.

En cliquant sur la deuxième boîte de *message* (**0 5000**), la sortie de *line* sera renvoyée à **0** plus de **5 000** millisecondes (5 secondes). En cliquant sur la première boîte, nous revenons à **100**.

La troisième boîte de *message* est un peu plus intéressante. Il utilise une fonction spéciale de la boîte de *messages* pour créer des messages multiples. Lorsqu'une **virgule** sépare les valeurs, elles sont traitées comme des messages distincts et fonctionnent comme si deux messages étaient envoyés l'un après l'autre avec le même cordon de raccordement. Dans ce cas, la valeur **20** est envoyée en tant que premier message. Comme il n'y a pas d'appariement temporel, la valeur actuelle de *line* passe immédiatement à **20**. Immédiatement après, la paire **50 2500** est envoyée, ce qui fait glisser la sortie de **20** à **50** en **2,5** secondes. Nous pouvons également utiliser la boîte de *nombres* pour faire sauter immédiatement *line* à une nouvelle valeur; étant donné qu'aucune valeur temporelle n'est fournie, cela fera également sauter l'état actuel à la valeur saisie.

Le *patcheur* nommé **exemple 3** est la procédure la plus complexeque nous avons vue jusqu'à présent. Elle dépend du mécanisme **idl** de l'objet *lcd* effectuer un dessin procédural. Lorsque le message **idl** est activé (avec le message **idl 1**), l'objet *lcd* émet la position actuelle de la souris vers l'écran de l'objet *lcd,* lorsque vous déplacez votre souris dessus. Ces coordonnées X / Y sont envoyées sous forme de liste à la deuxième sortie de l'écran *lcd*.

Le cordon de connexion de couleur bleu clair sortant de l'écran lcd connecte les coordonnées de la souris à un objet *unpack* situé en haut du patch 3. Ces valeurs sont envoyées à notre code de dessin qui crée un ensemble de valeurs utilisées pour dessiner une paire de formes rondes qui suivent le curseur. L'objet *line* est utilisé pour ralentir les modifications dans les formes et fait en sorte qu'elles se positionnent lentement sur l'emplacement actuel du curseur. Activez le message **idl** à l'aide de la boîte *toggle*, puis faites glisser la souris autour de l'objet *lcd* pour voir comment il réagit.

Il y a deux choses très importantes dans cette procédure. Tout d'abord, l'utilisation de l'objet *line* pour déplacer les valeurs de sortie sur une période d'une seconde (**1000** ms). Cela donne aux résultats une sensation plus organique en les faisant **interpoler** au fil du temps. Deuxièmement, nous utilisons l'objet *cartopol* (**car**tésien (**to**)vers **pol**aire) pour changer la position X / Y ("cartésienne") du curseur en une paire de coordonnées Distance / Angle ("Polaire"), utile pour dessiner des contenu dans un domaine circulaire. Une fois que les coordonnées polaires sont calculées, elles sont mises à l'échelle pour correspondre à la taille de la zone de dessin circulaire, puis utilisées dans le cadre d'une boîte de message complexe à multi-*message* pour créer les deux «yeux» situés au centre de l'objet *lcd*.

La boîte de *messages* qui est au bas du code envoie cinq commandes de dessin en séquence à l'écran *lcd*. Au début **clear** supprime le dernier dessin pour fournir une animation continue. Les deux suivants dessinent deux cercles de fond en utilisant la commande **paintoval** à des positions fixes (**100 100 140 140** et **180 100 220 240**). Les trois derniers arguments de **paintoval** définissent la couleur (noir: **0 0 0**). Les deux derniers messages utilisent les valeurs de l'objet *pack* pour dessiner des "pupilles" vertes sur les yeux. Les quatre valeurs fournissent les angles de départ et d'arrivée des pupilles utilisant le message **paintarc**.

Il peut être difficile de comprendre toutes les fonctionnalités de cette procédure, mais cela en vaut la peine - c'est un autre exemple d'utilisation de la programmation mathématique pour créer un résultat final complexe et réaliste. Les graphiques simples en temps réel sont constitués de blocs de code procédural, comme celui-ci; prendre un ensemble de valeurs d'entrée et les mettre à l'échelle selon vos besoins est essentiel pour de nombreuses applications dans Max.

Résumé

Nous avons vu une programmation assez complexe qui fournit une fonctionnalité de dessin contrôlé dans un objet *lcd* en créant des boucles (*counter*), en mettant à l'échelle des valeurs d'entrée en valeurs de sortie (*scale*) et en interpolant des nombres dans le temps (*line*). Nous avons également travaillé avec quelques objets qui calculent la trigonométrie. En combinant ces nouveaux objets, nous pouvons exercer un contrôle procédural sur ce que nous voulons accomplir dans nos programmes.