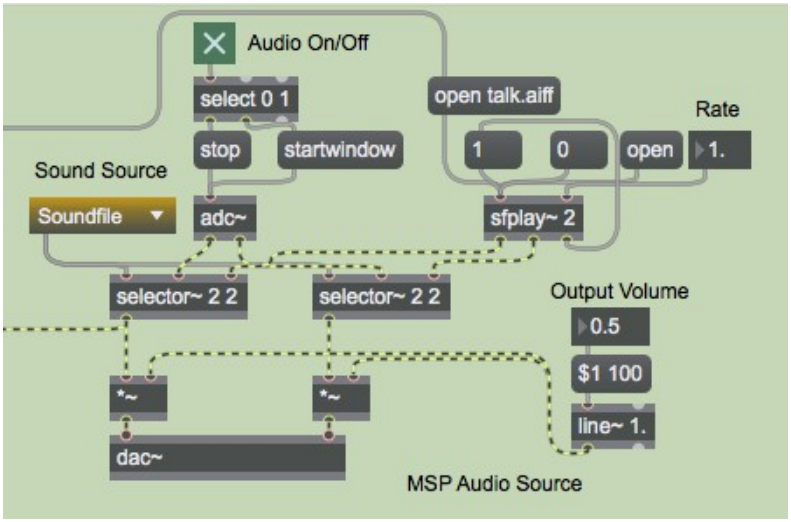
28-Contrôle audio de la vidéo

L'audio comme source de contrôle

Ce didacticiel montre comment suivre l'amplitude d'un signal audio MSP, comment utiliser l'amplitude suivie pour détecter des événements discrets dans le son et comment appliquer ces informations pour déclencher des images et contrôler des effets vidéo.

Dans le coin supérieur droit du patch, nous vous avons fait en sorte qu'il soit facile pour vous d'essayer l'une des deux sources audio: l'entrée audio de l'ordinateur ou un fichier son pré-enregistré.

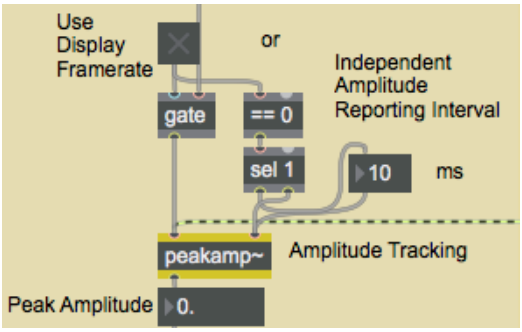


Le menu contextuel vous permet de sélectionner l'une des deux sources audio: adc~ ou sfplay~.

Nous avons utilisé un objet *loadbang* (dans la partie supérieure centrale du patch) pour ouvrir un fichier son AIFF ***talk.aiff*** et un film ***dish.mov***, et pour initialiser les paramètres des objets de l'interface utilisateur avec un *preset*. Ainsi, dans l'exemple ci-dessus, l'*umenu* a déjà sélectionné l'objet *sfplay~* comme source sonore, le fichier son a déjà été ouvert par le message **open talk.aiff**, le taux de *sfplay~* a été défini sur **1** et le volume de sortie a été défini sur **0,5**. Le canal gauche de la source sonore (la sortie gauche de l'objet *selector~*) est connecté à une autre partie du patch, qui suivra l'amplitude du son.

Suivi de l'amplitude de crête d'un signal audio

Pour suivre l'amplitude du son afin de l'utiliser comme donnée de contrôle dans Max, nous pourrions utiliser l'objet *snapshot~* pour obtenir l'amplitude instantanée du son, ou l'objet *avg~* pour obtenir l'amplitude moyenne du signal depuis la dernière fois où il a été vérifié, ou l'objet *peakamp~* pour obtenir l'amplitude de crête du signal depuis la dernière vérification. Nous avons choisi de suivre l'amplitude de crête du signal avec *peakamp~*. Chaque fois qu'il reçoit un **bang**, *peakamp~* rapporte la valeur absolue de l'amplitude de crête du signal qu'il a reçu dans son entrée gauche. Vous pouvez également le configurer pour qu'il rapporte automatiquement l'amplitude de crête à intervalles réguliers, en envoyant un intervalle de temps non nul (en millisecondes) dans son entrée droite, comme le montre l'exemple suivant.

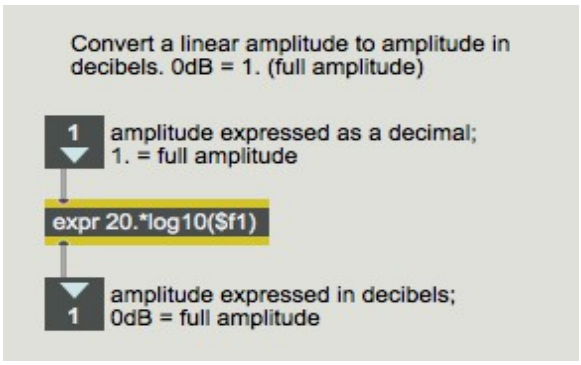


Un nombre non nul dans l'entrée de droite est un intervalle de temps de rapport en millisecondes

Toutes les 10 millisecondes, *peakamp~* enverra le pic d'amplitude de signal qu'il a reçu depuis le relevé précédent. Nous nous sommes donné la possibilité de désactiver la temporisation de *peakamp~* et d'utiliser le *metro* qui contrôle le taux d'affichage vidéo pour envoyer un **bang** à *peakamp~*, mais la capacité de temporisation intégrée de *peakamp~* nous permet de régler le temps de suivi audio indépendamment de le taux d'affichage vidéo.

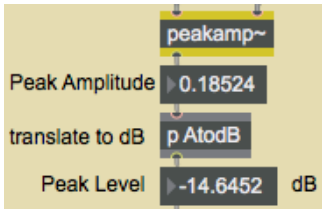
Utilisation de décibels

En fait, nous percevons l'intensité d'un son non pas tant comme une fonction linéaire de son amplitude, mais plutôt comme une fonction de son niveau relatif en ***décibels***. Cela signifie que plus de la moitié du niveau de pression acoustique que nous sommes capables entendre dans MSP se situe dans les 1% inférieurs de son amplitude linéaire, dans la plage entre 0 et 0,01! Pour cette raison, il est souvent plus approprié de traiter les niveaux sonores sur l'échelle logarithmique des décibels, plutôt que comme une valeur d'amplitude directe. Nous convertissons donc l'amplitude en décibels, en utilisant le sub-patch *p* **AtodB** (qui est identique à l'objet *atodb*).



Le contenu du sub-patch [**AtodB**]

Le sub-patch [***AtodB***] prend l'amplitude de crête signalée par *peakamp~* et la convertit en décibels, une amplitude de 1 étant égale à 0 dB et toutes les amplitudes inférieures ayant une valeur négative en décibels.



Convertir l'amplitude en décibels, par rapport à une amplitude de référence de 1.

Détail technique: La formule de conversion de l'amplitude en décibels est la suivante:

$$dB = 20 \cdot \log_{10}(\frac{A}{A_0})$$

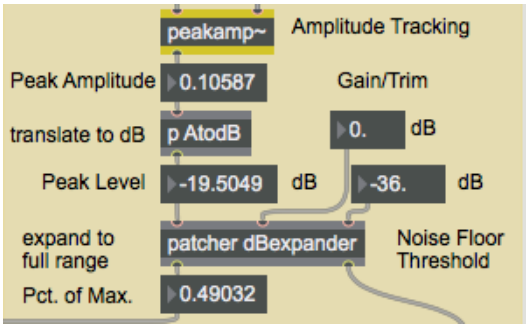
où A0 est une amplitude de référence et A est l'amplitude mesurée.

L'échelle des *décibels* est abordée dans les sections *Fonctionnement de l'audio numérique* et *Tutoriel MSP 4* des didacticiels MSP.

Se concentrer sur une gamme d'amplitudes

Dans de nombreux enregistrements et situations d'audio en direct, il y a beaucoup de sons de faible niveau que nous ne considérons pas vraiment comme faisant partie de ce que nous essayons d'analyser. Le son qui nous intéresse vraiment peut n'occuper qu'une certaine partie de la plage de décibels que MSP peut couvrir. (Dans certains enregistrements, la musique est compressée dans une plage extrêmement réduite pour obtenir un effet particulier. Même dans de nombreux enregistrements non compressés, les sons les plus importants peuvent tous se trouver dans une petite plage dynamique.) Le niveau du son faible et indésirable est appelé ***noise flor***. Ce serait bien si nous pouvions analyser uniquement les sons qui sont au-dessus de ce plancher de bruit.

Le sub-patch **dBexpander** nous permet de contrôler le niveau en dB de l'amplitude suivie et de définir un ***seuil de bruit de fond*** en dessous duquel nous voulons ignorer le signal. Le sub-patch prend les niveaux que nous ***voulons*** utiliser et les étend pour remplir la plage complète de l'échelle de décibels de 0 dB à –120 dB. Dans l'exemple suivant, nous avons spécifié un seuil de bruit de fond de –36 dB. L'amplitude du signal MSP à cet instant est de 0,251189, ce qui correspond à un niveau de –12 dB. Le sub-patch étend ce niveau (à l'origine –12 dans la plage de 0 à –36) afin qu'il occupe une position comparable dans la plage de 0 à –120. Le niveau résultant est de –40 dB, qui est envoyé par la sortie droite du sub-patch. Le niveau relatif au plancher de bruit de fond est envoyé par la sortie gauche, exprimé sur une échelle de 0 à 1, ce qui est une plage de contrôle utile dans Jitter. Dans cet exemple, le niveau d'entrée de –12 dB est supérieur de 24 dB au bruit de fond; c'est-à-dire qu'il est au 2/3 du maximum dans la plage de 36 dB spécifiée.



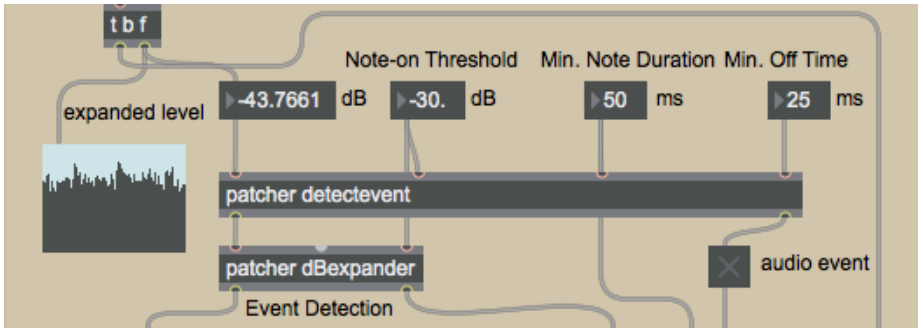
Convertir l'amplitude linéaire dans la région au-dessus de -36 dB en gamme complète.

• Vous pouvez appliquer cette valeur comme donnée de contrôle pour Jitter. Activez le *toggle* intitulée ***Use Display Framerate***. Cela désactivera temporairement le timer interne de *peakamp~* et utilisera le **bang** du *metro* à la place. Activez le *toggle* intitulé ***Audio On/Off*** pour lancer le traitement audio de MSP. Cliquez sur la boîte de *message* contenant le chiffre **1** au-dessus de l'objet *sfplay~* pour démarrer la lecture du fichier son. Activez le *toggle* intitulé ***Display Movie*** pour lancer la lecture de la vidéo. L'amplitude de crête du son est signalée au même rythme que l'affichage de la matrice vidéo – soit toutes les 25 millisecondes. Le niveau de décibels suivi - 40 valeurs par seconde – s'affiche dans le *multislider* vert et noir intitulé ***expanded level***. Le niveau, mappé dans la plage de 0 à 1, est utilisé pour modifier l'attribut **val** de l'objet *jit.op*, ce qui affecte la vidéo affichée. Vous pouvez augmenter ou diminuer la plage de cette valeur à l'aide de la boîte de *nombre* intitulée

Effect Strength. Les valeurs comprises entre 0,5 et 1,5 ont le plus d'effet sur l'image.

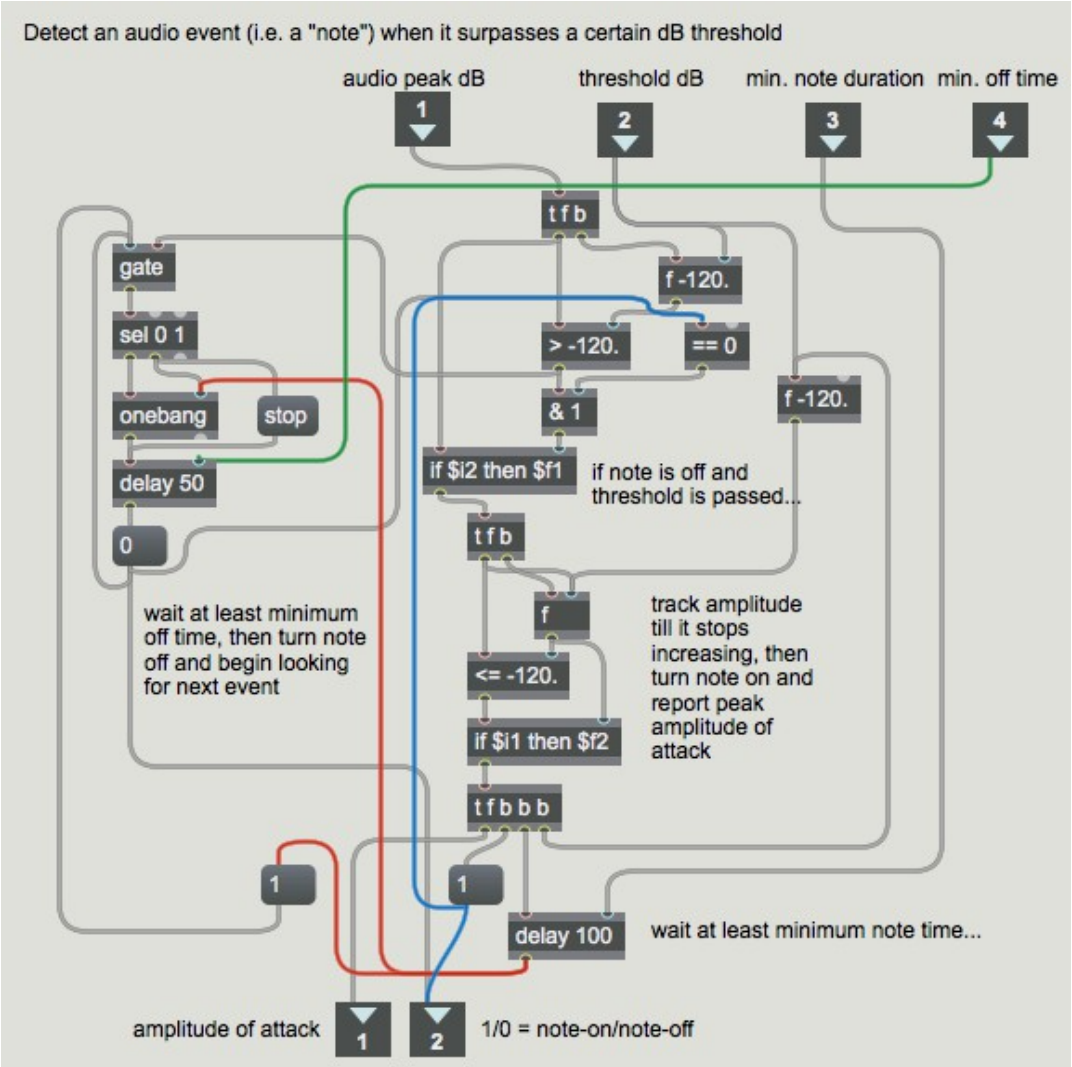
Détection d'événements audio

Dans la section précédente, nous avons suivi l'enveloppe d'amplitude du son et utilisé l'amplitude maximale pour obtenir une nouvelle valeur de contrôle pour chaque image de la vidéo. Nous pouvons également analyser le son à un niveau structurel différent, en suivant le rythme d'évènements individuels dans le son: notes dans un morceau de musique, mots dans le texte parlé, etc. Pour ce faire, nous devrons détecter le moment où l'amplitude augmente au-delà d'un seuil particulier, ce qui signifie l'attaque du son, et le moment où le son est passé sous du seuil pendant un temps suffisant pour que l'événement soit considéré comme terminé. Nous faisons cela dans le sub-patch ***detectevent***. Dans le patch principal, nous fournissons trois paramètres pour le sub-patch [***detectevent***]: le ***Note-on Threshold*** (le niveau au-dessus duquel le son doit s'élever pour désigner un événement ou une note), le ***Min. Note Duration*** (un temps que le sub-patch attendra avant de rechercher un niveau qui repasse sous du seuil), et le ***Min. Off Time*** (le temps pendant lequel le niveau doit rester en dessous du seuil pour que la note soit considérée comme terminée). Dans l'exemple suivant, un événement de note sera signalé lorsque le niveau dépasse –30 dB, et la note ne sera considérée comme terminée que lorsque le niveau reste en dessous de –30 dB pendant au moins 25 millisecondes. Comme le sub-patch attendra au moins 50 ms avant même de commencer à chercher un niveau de fin de note, la durée totale de chaque note sera d'au moins 75 millisecondes.



Lorsque le niveau dépasse le seuil et atteint un maximum local, un événement audio est signalé.

• Pour voir le contenu du sub-patch, double-cliquez sur l'objet patcher [***detectevent***].



Détection d'événements basée sur une amplitude dépassant un seuil

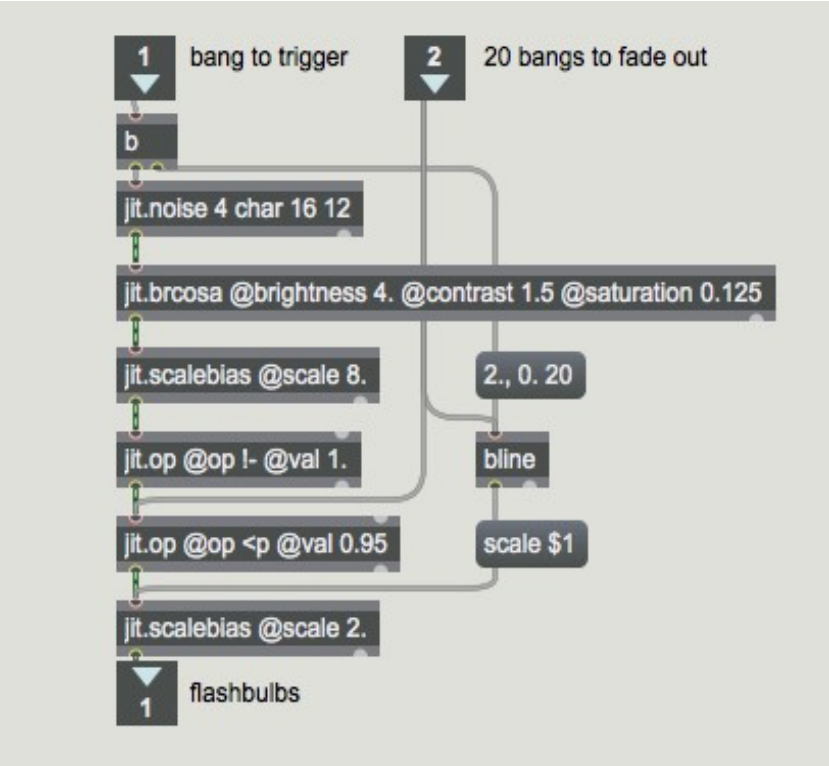
Les commentaires dans le sub-patch expliquent la procédure de manière assez succincte. Lorsqu'un nouveau niveau arrive dans l'entrée gauche, deux conditions doivent être remplies: le niveau doit être supérieur au seuil et il ne doit pas déjà y avoir une note en cours. Si ces deux conditions sont remplies, nous continuons à surveiller l'amplitude jusqu'à ce qu'elle cesse d'augmenter, et à ce moment-là, nous considérons que la note est complètement jouée, donc nous envoyons le chiffre **1** par la sortie droite et le niveau maximum par la sortie gauche. Nous attendons un ***temps de note minimum***, puis ouvrons la *gate* pour commencer à chercher des indications (provenant de l'objet *>)* signes que le niveau est passé sous du seuil. Une fois qu'un tel niveau a été détecté, nous attendons le temps minimum de désactivation avant de décider que la note est désactivée. Si un autre niveau supérieur au seuil survient avant que le temps minimum de désactivation ne soit écoulé, l'objet *delay* est **stop**(pé) et un nouveau niveau de désactivation doit être détecté. Lorsque la note est vraiment désactivée, un **0** est envoyé par la sortie droite, le fait que la note a été désactivée est noté (dans l'objet == **0**) et la *gate* est refermée. Elle est maintenant prête pour la prochaine fois que le seuil sera franchi.

• Fermez la fenêtre du sub-patch [***detectevent***]. Pour que ce détecteur d'événements fonctionne bien sur des sons qui changent rapidement, l'amplitude du pic doit généralement être suivie à un rythme assez rapide. Désactivez le *toggle* ***Use Display Rate*** pour que l'objet *peakamp~* utilise son timer interne à un intervalle de 10 ms.

Dans le patch principal, vous pouvez voir trois façons d'utiliser la sortie du sub-patch [***detectevent***]. Dans le coin inférieur droit du patch, nous utilisons le **1** de la sortie droite du *patcher* **detectevent** pour déclencher un autre sub-patch, le *patcher* **flashbulbs**, qui place des points colorés aléatoires dans une fenêtre d'affichage. Nous prenons la valeur de la sortie gauche du *patcher* **detectevent** et étendons sa plage comme nous l'avons fait pour le niveau audio d'origine, de sorte que la valeur signifiant l'amplitude de la note puisse couvrir toute la plage disponible. Nous l'utilisons pour déclencher des notes MIDI, et aussi pour choisir différentes images à afficher. Examinons brièvement chacune de ces procédures.

Utilisation des informations sur les événements audio

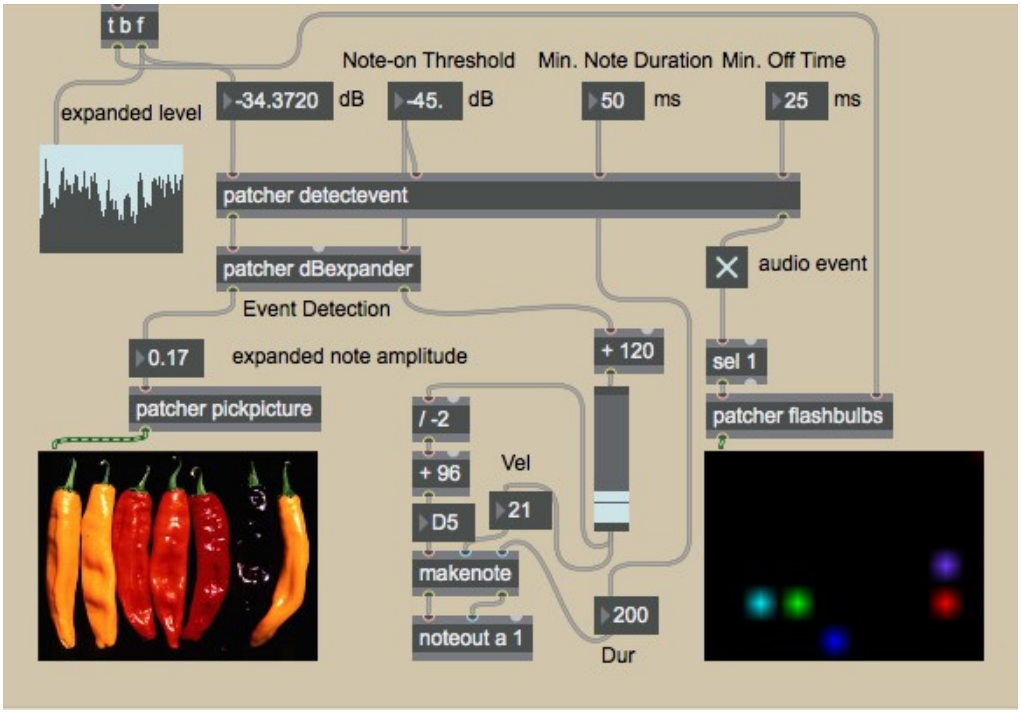
L'utilisation la plus simple d'un événement audio consiste à déclencher quelque chose lorsqu'un événement se produit. Chaque fois qu'un événement audio est détecté, nous déclenchons le sub-patch **flashbulbs**. Ce sub-patch génère une matrice 16x12 de couleurs aléatoires, puis utilise la mise à l'échelle pour transformer la plupart des couleurs en noir, ne laissant que quelques cellules restantes en couleur. Lorsque cette matrice est transmise vers le patch principal, ces cellules sont sur-échantillonnées par interpolation dans la fenêtre *jit.pw* et ressemblent à des éclairs de lumière colorée. Les valeurs de niveau inférieures de *peakamp~* sont utilisées dans le sub-patch [***flashbulbs***] pour frapper d'un **bang** un objet *bline*, ce qui fait que les couleurs s'estompent après 20 **bangs**.



Le sub-patch [***flashbulbs***]

Dans le sub-patch **pickpicture**, nous divisons simplement les amplitudes des événements en cinq parties égales et utilisons ces valeurs pour déclencher l'affichage de l'une des cinq images différentes.

Dans l'exemple suivant, vous pouvez voir l'utilisation des informations audio pour déclencher des notes MIDI.



Le niveau de crête détermine la hauteur et la vélocité d'une note MIDI

Nous utilisons la valeur de décibels expansée sortant de la sortie droite du *patcher* **expander** pour dériver des valeurs de hauteur et de vélocité MIDI. Nous plaçons d'abord les valeurs dans la plage 0 à 120, puis nous utilisons ces valeurs comme des vélocités MIDI et nous les mappons également dans la plage 96 à 36 pour les utiliser comme numéros de touches MIDI. (Notez que nous inversons la plage afin d'attribuer les événements les plus forts aux notes MIDI les plus basses plutôt qu'aux plus hautes, afin de leur donner plus de poids musical.) Les durées des notes peuvent être définies par la boîte de *nombre* ***Min. Note Duration***, ou elles peuvent être définies indépendamment en entrant une durée dans la boîte de *nombre* juste au-dessus de l'entrée de durée de *makenote*.

• Vous pouvez expérimenter davantage avec ce patch de plusieurs façons: en modifiant le ***taux*** du fichier audio avec la boîte de *nombre*, en ouvrant différents fichiers sons et films, en choisissant ***Sound Imput*** dans l'*umenu* pour utiliser une entrée de son en direct, et en changeant les différents paramètres de suivi tels que ***Reporting Interval***, ***Noise Floor Threshold***, ***Note-on*** et ***Min. Note Duration***.

Sommaire

Nous avons montré comment suivre l'amplitude maximale d'un son avec *peakamp~*, comment convertir l'amplitude linéaire en décibels, et comment détecter des événements audio en vérifiant si le niveau d'amplitude a dépassé un certain seuil. Nous avons utilisé les informations que nous avons obtenues de l'amplitude et les événements de crête pour déclencher des images de manière algorithmique, choisir parmi des images préchargées, jouer des notes MIDI et modifier des effets vidéo.