```
(
//Artificial Space, First Realization -- SuperCollider Code
// single channel processing interface
// hi pass/lo pass/ring mod/delay/limiter
// jesse pearlman karlsberg 31 october, 13, 15 december 2002

//input channel three
//high and low pass filters with ring modulator, delay, & compressor
                          //similar patches exist for audio channels 5
                          //and 7, differing only with respect to their
                          //pan settings and input channel

//DESCRIPTION
//this program allows for various combinations of digital signal
//processing of a single channel of audio input. the processing may
//include high and low pass filters with variable frequencies, a ring
//modulator with variable modulation frequency or modulation
//frequency assigned to pitch following, a delay of the input signal
//in addition to other processing, and a limiter with variable
//threshold. the program sends the output signal through two azimuth
//panners. one goes to a fixed pan position corresponding to one or
//both of a pair of loudspeakers. the other can be focused on any of
//the third through fifth output channels (directed to headphones)
//with a width ranging from one to three channels wide.

//PERFORMANCE INSTRUCTIONS
//a performance of artificial space is in three movements. movements
//should be approximately the same length, but that length may vary
//according to the performance situation. processing interfaces for
//all three input channels (connected to microphones) should be run
//simultaneously, and the output channels should be patched to a pair
//of loudspeakers (chans 1 and 2), and three headphones, one paired
//with each microphone (chans 3, 5, and 7). each processing interface
//has a corresponding set of presets (one for each section) which
//should be loaded prior to a performance, and cycled through prior
//to each section. the presets set the initial processing for that
//section and the pan positions for that section, but the processing
//(though not the pan) may be changed during a section. the three
//performers in addition to the processor should wear the headphones
//for the duration of the performance and improvise in response to
//the sounds they hear over the headphones.

//THE PRESETS
//each set of presets contains three presets. the first set is panned
//to the headphone corresponding to the input channel with some ring
//modulation and a 1 second delay. the second set is panned to all
//three headphones with a long delay and no other processing. the
//third set is panned like the second set with ring modulation, some
//pitch following, and a 1 second delay.

SC.chans = 5;       //five channels of output
Preset.funcInit({arg items, lpfreq, hpfreq, rmfreq, delay,
                    gainin, delvol, gainout, onoff1, onoff2,
                    thresh, panpos, width, scope = \ScopeV;

       items.name_("processing for first microphone");

       items.setItems(
              lpfreq       .freq(20000)        .name_("low pass "),
              hpfreq       .freq(20)           .name_("high pass"),
              rmfreq       .freq(200)          .name_("ring mod "),
              delay        .sp(1, 0.1, 10)     .name_("delay tm "),
              gainin       .db(0)              .name_("gain in  "),
              delvol       .db(0)              .name_("delay vol"),
              gainout      .db(0)              .name_("proc vol "),
```

```
		onoff1		.sp(0, 0, 1)		.name_("rm on/off"),
		onoff2		.sp(0, 0, 1)		.name_("pf on/off"),
		thresh		.db(15)			.name_("threshold"),
		panpos		.sp(0.8, 0.8, 1.6)	.name_("pan position"),
		width		.sp(1, 1, 3)		.name_("pan width   "),
		scope
	);

	items.sound_({
		var in, pitch, haspitch, hilo, out;

		//the input
		in = AudioIn.ar(3, gainin.kr);   //input from channel 3
		#pitch, haspitch = Pitch.kr(in); //pitch following

		//the processing--
		//high and low pass filters and ring modulator
		hilo =
			(LPF.ar(		//low pass filter
				HPF.ar(in, hpfreq.kr),	//high pass filter
				lpfreq.kr
			) * (1 - onoff1.kr))
			+
			(LPF.ar(		//ditto
				HPF.ar(in, hpfreq.kr),
				lpfreq.kr
			) * (SinOsc.ar(		//with ring modulator
					//slider controlled
					(rmfreq.kr * (1 - onoff2.kr))
					//pitch-follow controlled
					+ (pitch * onoff2.kr)
				)) * onoff1.kr);

		out = DelayL.ar(		//delay of input
			//delayed input times out volume
			in * delvol.kr, 10, delay.kr,
			//processed sound times out volume
			1, hilo * gainout.kr
		);

		out = Compander.ar( //compander...
			out, out,
			thresh.kr, 1, 0		//...as a limiter
		);

		//output
		Scope.ar(
			scope.myView,
			PanAz.ar(		//pan to headphones
				SC.chans, out,
				//variable pan position and width
				panpos.kr, 1, width.kr
			) +
			PanAz.ar(		//pan to loudspeakers
				SC.chans, out,
				//mono pan on channel 1 loudspeaker
				0, 1, 1
			)
		);
	});
}).show
)
```