A New Interface for Cloning Objects in Drawing Systems

Loutfouz Zaman, Wolfgang Stuerzlinger

York University, Toronto, Canada

ABSTRACT

Cloning objects is a common operation in graphical user interfaces. One example is calendar systems, where users commonly create and modify recurring events, i.e. repeated clones of a single event. Inspired by the calendar paradigm, we introduce a new cloning technique for 2D drawing programs. This technique allows users to clone objects by first selecting them and then dragging them to create clones along the dragged path. Moreover, it allows editing the generated sequences of clones similar to the editing of calendar events. Novel approaches for the generation of clones of clones are also presented.

We compared our new clone *creation* technique with generic duplication via copy-and-paste, smart duplication, and a dialog driven technique on a standard desktop system. The results show that the new cloning method is always faster than dialogs and smart duplication for most conditions. We also compared our clone *editing* method against rectangular selection. The results show that our method is better in general. In situations where rectangle selection is effective, our method is still competitive. Participants preferred the new techniques overall, too.

KEYWORDS: Cloning, object duplication, editing groups of objects

INDEX TERMS: H.5.2 [Information Interfaces and Presentation]: User Interfaces – Interaction Styles (e.g. direct manipulation).

1 INTRODUCTION

One of the key features of calendar applications is the creation and modification of repeated events. Repeated events are typically created via a dialog, where the user specifies the parameters of the repetition such as title, time, duration, intervals, and the last repetition, if applicable. After such a repeated event is created, the user can modify the entire series or individual events within the series. For this, the user has to select an event from the repeated series and modify the attributes of the repetition via a dialog. After the modifications are complete, a pop-up will ask the user which of the events to change. See e.g. Figure 1, which shows the pop-up from Google Calendar. The following options are commonly provided: change all events in the series, change only the current instance, or change all following events, i.e. all subsequent events.

Our fundamental idea is to apply this notion of editing sequences to the 2D drawing domain. Repeated events in a calendar can be considered to be "live" copies, since changes to the attributes of one of the events are *usually* applied to other events. In the graphics domain, the equivalent is a set of shapes that are clones of each other, such as a row of rectangles. Considering their order of creation, e.g. from left to right, or from top to bottom, such clones are then similar to a series of events in an ordered sequence. The similarity is even more apparent, if the objects were cloned via copy & paste. Changes to the physical appearance, such as shape or color, are then similar to changing event attributes, such as time. Editing all following events can be thought of as modifying clones created after a given clone.

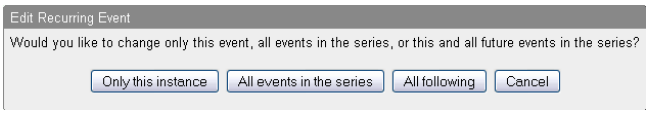

Figure 1. Editing recurring events in Google Calendar.

In this paper, we present Skencil-C, a mouse-based system which implements a new interface to cloning objects in 2D graphics environment, based on the paradigm discussed above. We evaluated the new techniques against existing techniques in two user studies. To our knowledge, this is the first evaluation of cloning techniques in 2D drawing systems.

2 MOTIVATION AND PREVIOUS WORK

There are four basic principles of graphic design: *contrast*, *repetition*, *alignment*, and *proximity* [13][18]. *Repetition* means reuse of the same or similar elements throughout a design and ensures a sense of unity, consistency and cohesiveness. *Alignment* ensures that elements in a design are not placed randomly – every element is connected via invisible lines, even far apart elements. *Proximity* is about moving things closer or farther apart to achieve an organized look, and groups related items. *Contrast* emphasizes elements. The concept of *rhythm* is related to repetition and repeats similar elements with a variety of forms or spatial intervals [12]. 3D design also involves lots of repetition and alignment [3][4]. The novel cloning tool presented here supports at least two of these principles (*repetition* and *alignment*) directly and makes operations related to them more efficient. This allows graphic designers to use cloning as a primary mechanism in their work.

The fundamental difference between cloning and copying is that copying creates a set of independent objects while cloning retains information about the cloning structure, i.e. history. Today, cloning facilities are available in popular 2D and 3D applications, including calendars, e.g. Google Calendar, 2D drawing systems, e.g. Inkscape, and 3D drawing systems, e.g. Autodesk 3DS Max. Cloning is also frequently used in spreadsheets, through formulas. Fujima *et al.* [8][9] generalized the notion of cloning in spreadsheets to enable seamless connections between web applications. However, the cloning user interface in the above-mentioned applications is indirect, mostly through dialogs.

The issue of context-awareness for copy-paste operations is another motivation for our work, as such systems retain additional information, e.g. about the object or previous operations, and use this information to improve the user interface. Citrine [17] extends the copy-and-paste paradigm by storing the type of data on a copy and then performing different operations on paste. The idea of a clipboard extension via a malleable physical interface allows each clip to have its own dedicated control key [2]. CorelDraw and

4700 Keele Street, Toronto, www.cse.yorku.ca/~wolfgang

Adobe Illustrator CS4 include a smart duplication technique, where the user can copy and paste an object repeatedly [5]. The relative offset of the first copy is applied at each subsequent paste. This makes it easy to create a row of objects, but each copy still requires an individual action. Microsoft Word and PowerPoint 2007 include this as well, but the feature is missing from the 2008 version. Some of these packages feature snapping to align new duplicates to previously inserted objects. However, even with alignment, multiple actions are required to create multiple copies.

The Fill Interpreter [14] is an early system that unifies region filling, brushing and compositing in a graphical programming language. Filling is a different approach to object replication, which does not preserve the identity of the original object(s).

Our new cloning technique enables the creation of clones from a selected object through a drag operation with a modifier key. Creation of (adjacent) *copies* of objects through dragging has previously been presented for 3D environments in the MIVE System [16]. One downside of this system is that the structure of the copy operation is not retained. Chen *et al.* [3][4] also studied interfaces for cloning in virtual immersive environments. They developed six cloning interfaces and compared their usability in structural engineering tasks. The new techniques resulted in significant performance gains and a better workflow.

Single object selection in graphical user interfaces is a very well studied task. Selection of more than one object is related to our work, as it is an alternative for manipulating a set of cloned objects. Multiple object selection is often accomplished via rectangle selection with the mouse or lasso selection with a pen. Novel group selection techniques, based on the Gestalt principle, scale to larger object groups and work even in non-rectangular layouts, see e.g. [6][7]. The Handle Flags selection technique displays potential selections that an ink stroke or group could belong to and allows the user to access desired item(s) [10]. However, most of this work targets pen-based systems and is not directly applicable to mouse-based systems. Moreover, selection alone is not sufficient for clone editing, as a selection operation does not provide information about the cloning structure, nor does it guarantee that the selection aligns with that structure.

Alignment is also related to our work. The alignment stick is a post-hoc alignment technique where objects are aligned by being pushed by the alignment "stick", based on the ruler metaphor [15]. Snap-dragging [1] allows the user to draw new alignment objects. Thus, it becomes possible to align precisely along curves, not just the coordinate axes. Constraint editors also support alignment and have been studied extensively. They are not widely used, because solving constraints is not easy and, more importantly, users do not want to define and maintain constraints for simple drawing tasks [11]. Many drawing applications provide snapping to grids for object alignment. HyperSnapping [11] interactively changes the spacing of the snapping grid while an object is dragged.

3 CREATING CLONES

In our system clones can be created either along a free form path or along a linear path. The corresponding user interface techniques are named *freeform direct cloning* and *linear direct cloning*. All operations discussed in the following are integrated with undo and other common user interface mechanisms, as applicable.

3.1 Freeform Direct Cloning

To create clones along a free path, a single object has to be selected first via normal selection techniques. Dragging the object while holding down a modifier key, <WINDOWS>, then activates cloning. A static outline representing the proposed location of a clone appears each time the distance between the current and the previous outline (or the original object) exceeds a threshold, see Figure 2a. We currently use 20% of the diameter of the bounding box of the object. The last outline with the four-arrow cursor moves with the mouse. After the user releases both the left mouse button and the <WINDOWS> key, the direct cloning action is complete and the proposed clones are instantiated as clones of the object in the document, see Figure 2b. In effect, the selected object is replicated a number of times proportional to the length of the cursor drag. We call this *freeform direct cloning*. To aid the user, the current clone count is displayed in the status bar area during dragging. *Drag-and-release* duplication is a related technique, e.g., in Adobe Illustrator CS4. However, only a *single* duplicate can be created with drag-and-release duplication at a given time. Creation of multiple duplicates still requires multiple actions.

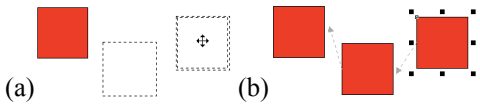

Figure 2. Freeform direct cloning.

The arrows that appear upon completion illustrate the order in which the clones were created. They point to the object that was created just before. These arrows are only shown when any object in a set of clones is selected to visualize the entire sequence.

3.2 Linear Direct Cloning

Because horizontal and vertical arrangements are very common, the system also supports creation of clones along a straight path in either dimension. This technique is called *linear direct cloning*. Holding the <ALT> key in addition to the <WINDOWS> key and dragging a selected object activates this mode. Then, the system assists the user by snapping a straight path to one of the four main directions. The direction is chosen based on the predominant direction of the moving mouse cursor. Figure 3 illustrates an example when the cursor is moved to the right, and the outlines and the resulting clones are aligned horizontally.

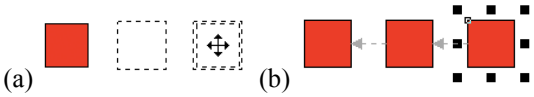

Figure 3. Linear direct cloning.

3.2.1 Retraction

Retraction was inspired by the interactive table insertion interface of Microsoft Word. It can be activated when the system is in the linear direct cloning state and the mouse cursor is moved towards the original object, i.e. in the direction opposite to the initial movement direction. Then, clone outlines are deleted one by one until the selected object is reached. Figure 4 illustrates this.

3.2.2 Change of Spacing

Spacing can be changed in two ways: by changing the number of proposed clones or by shifting each outline proportional to the change in position of the last object. Both methods can be used in any order and multiple times within a single direct cloning operation. The first method allows the user to distribute clones differently by changing their number, while keeping the mouse cursor in the same position. This is accomplished by scrolling the mouse wheel, which increases or decreases the number of proposed clones incrementally, see Figure 5. Initially, two outlines are created, see Figure 5a. Scrolling the mouse wheel down results in a decrease in the number of proposed clones, see Figure 5b, which results two clones, see Figure 5c.

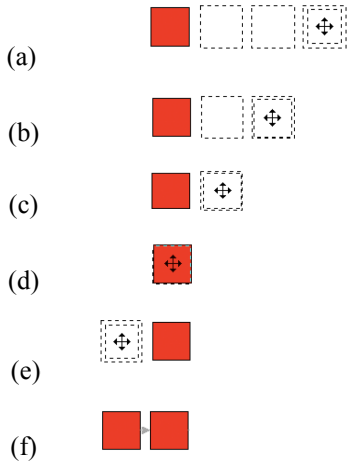

Figure 4. Outline retraction.

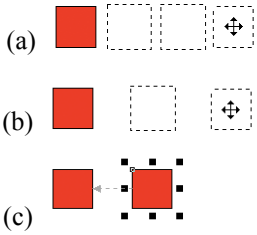

Figure 5. Changing spacing by changing the number of outlines.

In the second method, moving the mouse cursor changes the spacing between clones. In this case, the number of clones does not change, but the inter-clone distances vary. This mode is activated by releasing the <ALT> key while in linear direct cloning. Then, moving the cursor towards or away from the selected object, changes the spacing between the proposed clones, see Figure 6a,b. The minimum permitted distance between the proposed clones is zero, i.e. when objects touch each other. Releasing the mouse button generates the clones (Figure 6c).

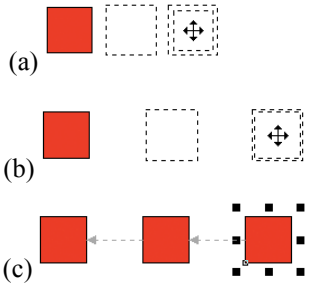

Figure 6. Changing spacing by moving the cursor.

3.3 Extending Sets of Clones and Clones of Clones

An existing set of clones can be extended by performing direct cloning on the last clone in the set after it has been selected, see Figure 7. However, an extension operation can be performed not only on the last object in a set, but on any other object as well, including the *master* object, i.e. the original, first object. This is referred to as creating *clones of clones*. Each object in a set of clones of clones can have infinitely many subsets of clones.

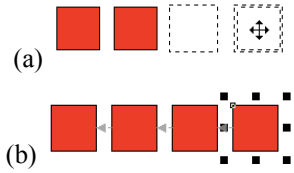

Figure 7. Extending an existing set of clones.

An example is illustrated in Figure 8. The middle object in a set of three clones is initially selected. Performing direct cloning will then result in a new set of clones, see Figure 8b,c. In another step, shown in Figure 8d, direct cloning is then performed on the middle object in the new, extended group. This yields another set of clones, see Figure 8e. Figure 8f will be discussed below.

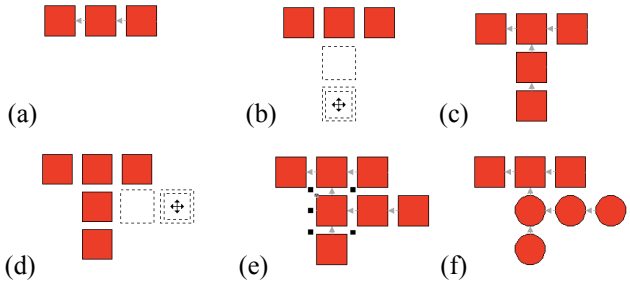

Figure 8. Creating and editing clones of clones.

3.4 Clones of Groups

Direct cloning can be performed not only on individual objects, but on groups as well. First, a group of objects must be selected. Direct cloning of this group will then generate a clone group. An example is illustrated in Figure 9. First, a set of clones is selected, see Figure 9a. Figure 9b shows an intermediate step in direct cloning on this group. Releasing the button instances the cloned group. Each object in the cloned group points to the corresponding object in the original group, see Figure 9c.

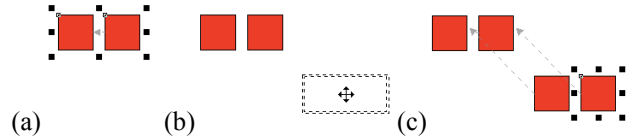

Figure 9. Clones of groups.

4 Editing Clones

In electronic calendars, the attributes of repeated events, such as time or location, can be changed for any event. The system then queries the user how to apply these changes via a pop-up dialog. Usually these changes can be applied to the selected event only, to all the events in the series, or to all the events following the chosen event. We implemented the same mechanism in our drawing system. When an attribute of a clone is changed, a dialog box pops up to ask how to apply this change, see Figure 10.

In most 2D graphics software, objects have several attributes that can be modified. The most common ones are position, orientation, size, color, and shape. Position is usually handled differently from the other attributes in that it can be modified only via direct manipulation. In our clone-editing interface, all the attributes except position can be edited, and changes can be replicated across clones in a set. This design decision was made to avoid user frustration by a pop-up every time a clone is moved.

Editing works also for clones of clones and clones of groups. An editing example is illustrated in Figure 8ef. First, an object in a set of clones of clones is selected. Then, its shape is changed from a square to a circle. A pop-up box is displayed after the completion of the edit. Assuming the user clicks on "All following", all objects created after or cloned from the selected object are changed, as illustrated in Figure 8f.

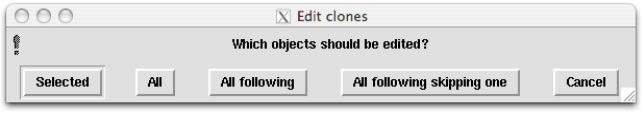

Figure 10. Edit clones pop-up box in Skencil-C.

4.1 Splitting Sets of Clones, Unlinking, and Deleting

Clone relationships can be broken at arbitrary objects. This results in two separate sets of clones. An example with a set of four clones is shown in Figure 11. First, the user right clicks on the object that will be a new master and selects "Unlink Remaining", see Figure 11a. This splits the set into two independent clone sets. Figure 11b and Figure 11c illustrate the resulting sets of clones.

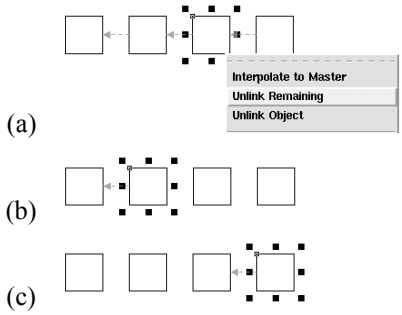

Figure 11. Splitting a set of clones.

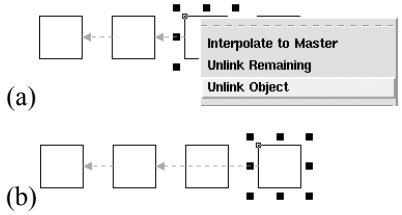

Figure 12. Unlinking a clone.

Clones can be unlinked from their sets. First, the object to be unlinked is selected and "Unlink Object" is selected in the right-click menu, see Figure 12. The system will then pop-up a dialog asking how to apply this change, see Figure 10. There are three choices. If the user chooses to change all objects, then all clones will be unlinked. If "All following" is selected, then all the objects including the selected one will be unlinked. Choosing "selected" will only unlink the selected one, see Figure 12b.

Clones can be deleted by selecting a clone and pressing the <DELETE> key, see Figure 13. Upon deletion of a selected clone, a pop-up asks how to apply the changes, similar to unlinking. It is then possible to delete all the objects, only the ones following the selected, or only the selected one as in Figure 13b. If a deleted clone has multiple branches, all branches are re-attached to the preceding object. For example, if the first changed clone is deleted in Figure 8f, both branches get attached to the object above it. If the deleted object is the root, we create independent clone sets.

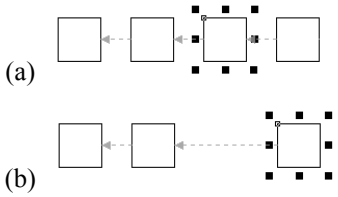

Figure 13. Deleting a clone.

5 EVALUATION

Our novel cloning interface was implemented by extending Skencil, an open source vector graphics editor written in Python. We named our system Skencil-C, and it implements all operations described in this document.

5.1 Pilot Study

We performed a preliminary user study to explore the performance of the direct cloning method. For this we re-implemented the tiled cloning user interface from Inkscape, where the user can specify repetition parameters in a dialog. That dialog contains fields for the number of repetitions (in one or two dimensions), for spacing, for size and/or color interpolation, as well as alternation, see Figure 14. The pilot mostly used earlier versions of the same tasks described in 5.4. One task compared free-form cloning vs. standard duplication. The study also featured one editing task similar to task 3 in section 5.8. However, that pilot task proved hard to analyze, as it involved both creation and editing.

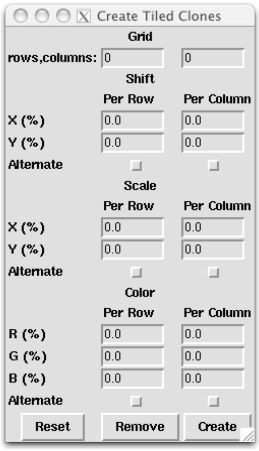

Figure 14. Tiled cloning user interface in Skencil-C.

The results of this pilot study showed that direct cloning is practically almost always faster than tiled cloning, which is not very surprising as finding the numeric values for various input fields is less intuitive than direct manipulation. Also, tiled cloning needed more "undo" operations (including re-creates) on average. We also found no significant difference between row and column creation. However, grid creation was not well supported at that time, due to the necessity to first duplicate objects into a row (or column) and then to duplicate that into the grid. Additionally, if the spacing of objects did not match preset values, tiled cloning was significantly slower, and also needed more "undo" operations. Direct cloning of objects along a free-form path is faster, but not always as accurate as direct placement. Unsurprisingly, all clone-based creation and editing operations on a set of cloned objects (such as creating and then coloring or interpolating) are faster in

general compared to tiled cloning. Finally, participants had a marked preference for direct cloning in general.

5.2 Modified Grid Creation Method

Based on the results of the pilot study, we implemented a new version of the grid creation method. This avoids the intermediate step of duplicating a row. Now, a single, diagonal, mouse drag directly creates a grid of clones, see Figure 15.

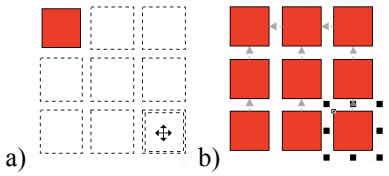

Figure 15. Creating a grid of clones in one operation.

While our design requires less actions from the user, we performed a new user study to compare against smart duplication and to see if the new designs translate into improved user performance and preference even for users with *little training*.

5.3 User Study 1

The goal of this user study was to compare *smart duplication* with the modified grid creation method and tiled cloning. Horizontal or vertical alignment was supported, via the <ALT> key. Participants were asked to use this feature during the study.

Twelve paid participants (10 males) were recruited. The participants were between 19 and 30 years with a median of 24. An average computer use of 48 hours per week was reported and 4.5 years of experience with either diagram editors or presentation software as well as 3.6 years with drawing applications. Two of the participants reported left-handedness, but chose to perform the experiment with the right hand. None of the participants had previous experience with cloning or Skencil-C.

The user study was conducted using a high-end laptop with a 22" external display at 1680×1050 in full-screen mode and a USB wheel mouse. All events and timings were logged.

5.4 Experimental Design

We used a 3×2×3 repeated measures design. Three tasks and two task sizes (small and large) were used to compare linear direct cloning vs. tiled cloning vs. smart duplication. In each trial, participants saw a master object designated by a red box and shaded target outlines designating the locations where the new clones needed were to be placed, see Figure 16. The measured dependent variable was task completion time. The set of tasks focused only on row and grid creation, because there was no evidence in the pilot study that row and column creation would be different. Task 1 featured row creation, while task 2 featured grid creation. The spacing of objects in both tasks matched the default preset for both direct and tiled cloning, whereas in smart duplication the first copy had to be positioned manually. Task 3 featured grid creation with non-preset spacing, i.e. the default value for object spacing was different than what was needed to perform the task. Task 3 was evaluated in a separate session because the pilot study revealed that participants tend to get confused when faced with too many similar conditions. Each participant preformed a single repetition. The two sessions were counter-balanced and the tasks, techniques, and sizes were also counter-balanced within each session. Control measures were used to prevent the participants from using the wrong technique.

At the end of the each session, participants were asked to rank each technique on a scale from 0 to 6. Moreover, participants were asked to provide an overall ranking for the techniques at the end.

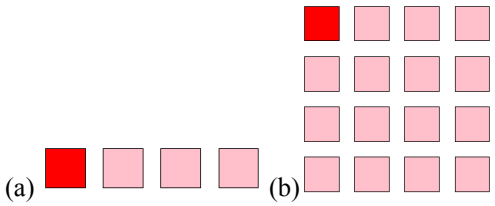

Figure 16. Tasks in user study 1, (a) task 1, (b) task 2 and 3.

5.5 Results

Overall, a repeated measures ANOVA shows that the main effects of technique, $F_{2,20} = 20.22, p < .0001$, task $F_{2,20} = 39.12, p < .0001$, and size $F_{1,10} = 38.13, p < .0001$ were significant, see also Figure 17. The interaction between technique and task was also significant, $F_{4,40} = 11.05, p < .0001$, see Figure 18. No ordering effects were observed, as session order had no significant effect, $F_{1,10} = 3.06, p > .05$. A Tukey-Kramer analysis was performed to detect task groupings. There was a grouping of tasks 1, 2 (fastest), and task 3 (slowest). The groupings were consistent with sessions. We report all further results according to this grouping.

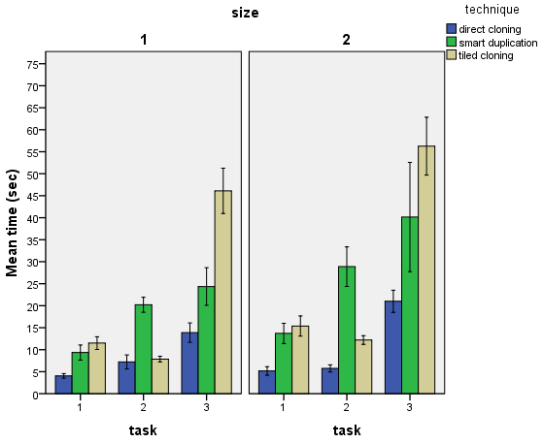

Figure 17. Average task completion times for User Study 1. Error Bars: ± 1 SE.

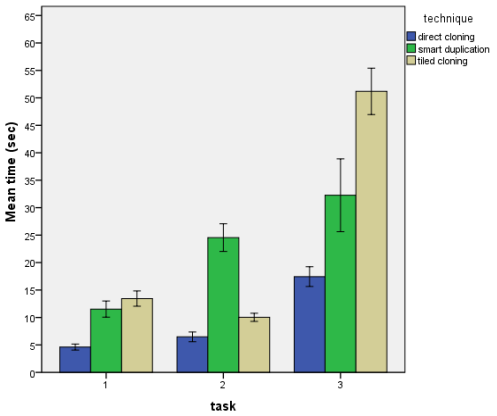

Figure 18. Main interaction between task and technique. Error Bars: ±1 SE.

5.5.1 Tasks 1 and 2

There was a significant difference for technique, $F_{2,20} = 48.26$, $p < .0001$, task, $F_{1,10} = 12.43$, $p < .01$ and size, $F_{1,10} = 10.51$, $p < .01$. A Tukey-Kramer analysis revealed all three techniques to be different, with direct cloning being the fastest and smart duplication the slowest. The interaction between technique and size was significant, $F_{2,20} = 4.77$, $p < .05$. Direct cloning for both size sets was faster than both size sets of smart duplication and faster than large tiled cloning tasks. The interaction between task and technique was also significant, $F_{2,20} = 16.14$, $p < .0001$.

5.5.2 Task 3

There was a significant difference for technique, $F_{2,20} = 12.03$, $p <. 0005$ and size, $F_{1,10} = 8.99$, $p < .05$. A Tukey-Kramer analysis revealed that direct cloning is faster than tiled cloning, but not compared to smart duplication. Although direct cloning and smart duplication were not significantly different, direct cloning was still faster on average (17.44 s vs. 32.25 s). The interaction with size was not significant, $F_{2,20} = 0.40$, ns.

5.6 Discussion

Direct cloning outperformed smart duplication when the spacing preset matched the desired distance. For a non-matching preset direct cloning was not significantly faster than smart duplication, but still faster on average. Based on the statistical power, we expect that this test may reach significance with more subjects. The study demonstrated also that the modified grid creation technique outperformed tiled cloning. Questionnaire analysis revealed that participants prefer direct cloning regardless of task as well as overall, see Figure 19.

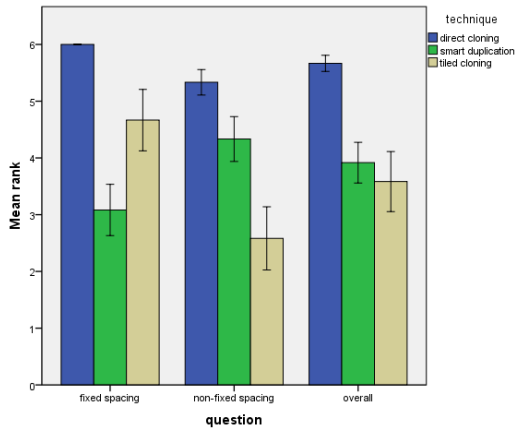

Figure 19. Questionnaire analysis for Study 1. Error bars: ± 1 SE.

We also compared some of the data from our pilot with this study for creating a row of clones with non-preset spacing. This was to compensate for the fact that we did not include such a condition in this first study. That comparison indicates that smart duplication is somewhat faster on average than direct cloning for creating clones with non-preset spacing. However, as this is a cross-study comparison with non-matching sets of tasks we cannot say if this difference is statistically significant.

5.7 User Study 2

Clone editing is another important aspect of our work. In the pilot study we saw evidence for its effectiveness. But we encountered problems since we mixed creation with editing in the corresponding task, which compromised internal validity. Hence, we focused only on editing in this second user study to get more precise information. All clones and their topologies were preloaded for each trial, so that participants didn't have to create them. The tasks then required the user to (re-)color clones with at most three colors. This tested if participants were capable of understanding an arrangement of clones and how quickly they could edit it. Another aspect was to investigate how clone editing compares to editing with rectangular selection, including the ability to add/subtract objects from the selection with <SHIFT>. As some participants in the pilot study had trouble understanding the structure of non-grid cloning arrangements, we extended Skencil to designate the master object by a dashed outline, see Figure 20.

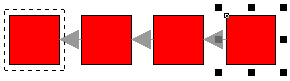

Figure 20. Master object designated by a dashed outline.

5.8 Experimental Design

We used a repeated measures design. Five tasks and two task sizes (small and large) were used to compare linear clone editing with editing with rectangle selection in a 5×2×2 design. All 20 conditions were counter-balanced. The measured dependent variable was task completion time. The participants and apparatus from user study 1 were also used for this study.

We used two different sizes. Size 1 tasks featured three clones counted from the master object or stem, and size 2 tasks seven clones. Tasks 1 and 2 required coloring a single branch, whereas tasks 3 to 5 required coloring all objects.

The layouts used in the tasks covered two topologies: all clones directly connected to a master object, such as a star arrangement, Figure 21b, or a main "stem" with branches to each node, such as a straight grid, Figure 21a,c, a "fishbone" grid, Figure 21d, and a star without center Figure 21e. In each layout the target colors alternated between red and blue every two columns/spokes. For all layouts except the straight grid, there were either thirteen "spokes" or "branches". In all tasks, the target configuration would appear on the left half of the screen for reference, while the editable clone topologies would appear on the right. In the rectangle selection condition, participants were asked not to use <SHIFT> in task 1, but were allowed to add/subtract objects with it in all other tasks, as necessary. For clone editing, participants were only allowed to select a single object at a time and to operate on it. Skencil-C shows the topology *only* in the clone editing condition, which helped participants to remember the correct editing technique.

At the end of the experiment, participants were asked to rank selection and clone editing from 0 to 6 for rectangular and non-rectangular arrangements, as well as overall.

5.9 Results

The main effects of the editing technique, $F_{1,11} = 9.52$, $p < .05$, task $F_{4,44} = 59.72$, $p < .0001$, and size $F_{1,11} = 15.55$, $p < .005$ were significant, see also Figure 22. The interaction between technique and task was also significant, $F_{4,44} = 20.12$, $p < .0001$, see Figure 23. The interaction between technique and size was significant, $F_{1,11} = 30.64$, $p < .0005$, and between task and size, $F_{4,44} = 7.52$, $p < .0005$. A Tukey-Kramer analysis revealed three groupings of tasks: tasks 1-2 (fastest), task 3 (medium) and tasks 4-5 (slowest). We report all results according to this grouping.

5.9.1 Tasks 1 and 2

Technique was not significant for these tasks, $F_{1,11} = 3.57$, $p > .05$. But size, $F_{1,11} = 21.23$, $p < .001$, and task were, $F_{1,11} = 23.97$, $p < .0005$. The interaction between technique and size was

significant, $F_{1,11} = 31.49, p < .0005$. Clone editing took roughly the same time for both sizes, but rectangle selection was slower for larger sizes. The interaction between task and size was significant, $F_{1,11} = 6.34, p < .05$, with task 2 slower for larger sizes.

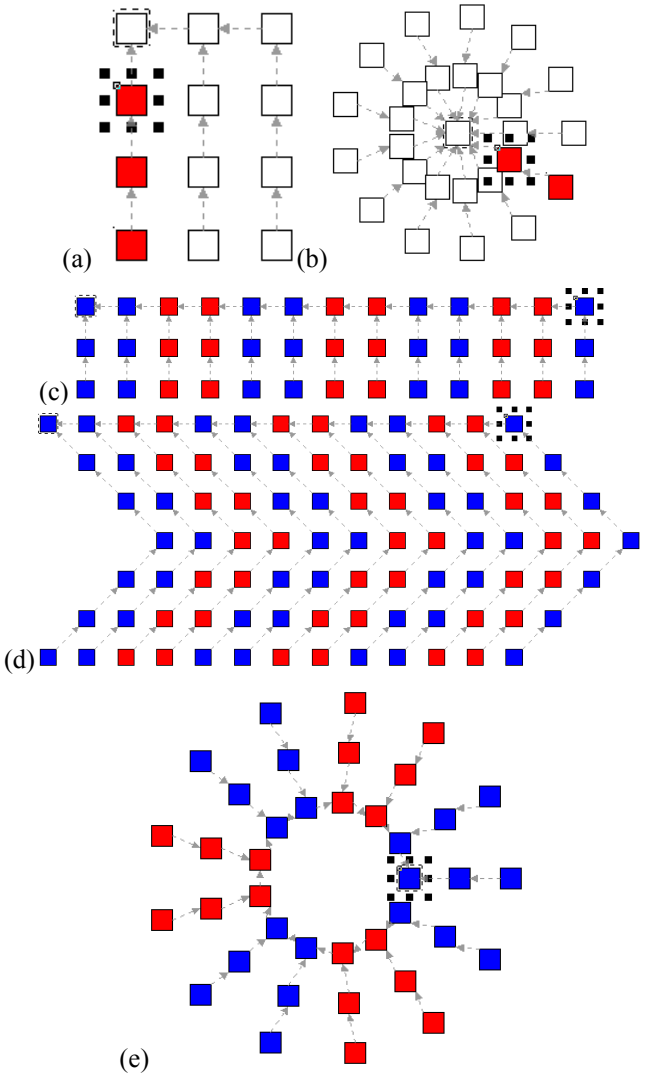

Figure 21. Task in second study: (a) task 1, size 1, (b) task 2, size 1, (c) task 3 size 1, (d) task 4, size 2, (e) task 5 size 1.

5.9.2 Task 3

Here, technique was significant, $F_{1,11} = 23.28, p < .001$. Rectangle selection was faster (30.04 s vs. 50.98 s). Size was not significant, $F_{1,11} = 0.43$, ns, and there was no significant interaction.

5.9.3 Task 4 and 5

For these, technique was significant, $F_{1,11} = 23.36, p < .001$. Clone editing was the fastest (52.74 s vs. 82.78 s). Task was not significant, $F_{1,11} = 0.11$, ns. The interaction between technique and task was significant, $F_{1,11} = 10.42, p < .01$. The differences for task 4 were much larger than for task 5. Size was significant, $F_{1,11} = 12.99, p < .005$. Larger sizes took longer (52.97 s vs. 82.55 s). The interaction between technique and size was significant, $F_{1,11} = 32.57, p < .0005$, with no difference for small tasks, but a big difference in larger sizes.

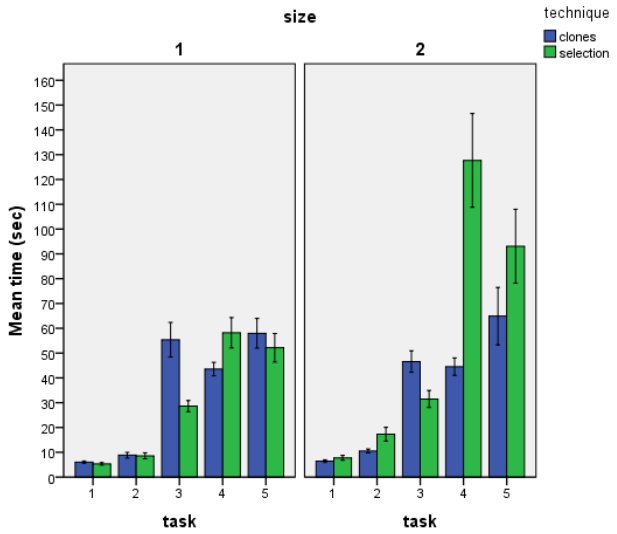

Figure 22. Average task completion times for User Study 2. Error Bars: ±1 SE.

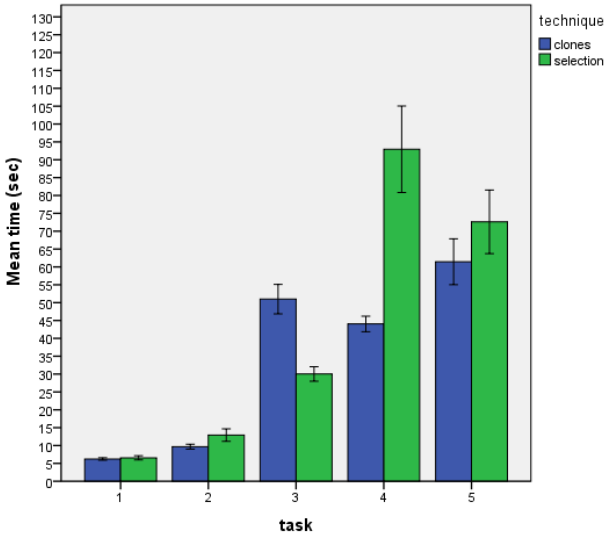

Figure 23. Interaction between task and technique in User Study 2. Error Bars: ±1 SE.

5.9.4 Questionnaires

The results from the questionnaires are shown in Figure 24. Overall, clone editing was preferred.

5.10 Discussion

Unsurprisingly, both techniques performed equally well for the simple tasks 1 and 2, where only a single "branch" was to be colored – as long as the number of objects was small. However, with larger numbers of objects, rectangle selection became significantly slower. We attribute this to the larger distances that the cursor must cover with this technique and to the effect of arrangements (such as the branches of the star), where rectangle selection cannot be used effectively.

Tasks 3–5 involved coloring all objects. It was possible to complete this task a bit faster by first coloring all objects one color and then coloring only half the objects differently. About half the participants employed this strategy at least once, while the other

half never did it. This seems to indicate different strategies in human problem solving. Task 3 was ideally suited for rectangle selection, and it is no surprise that this technique was consistently faster here. However, tasks 4 and 5 also show that as soon as the layout does not match the kind of tasks that rectangle selection supports well, the technique quickly becomes non-competitive. Interestingly, the large "fishbone" grid pattern used in task 4 was particularly challenging for rectangle selection, even more than so than the star pattern in task 5.

In line with the timings, the analysis of the questionnaires showed that participants prefer rectangle selection for rectangular arrangements and clone editing for everything else.

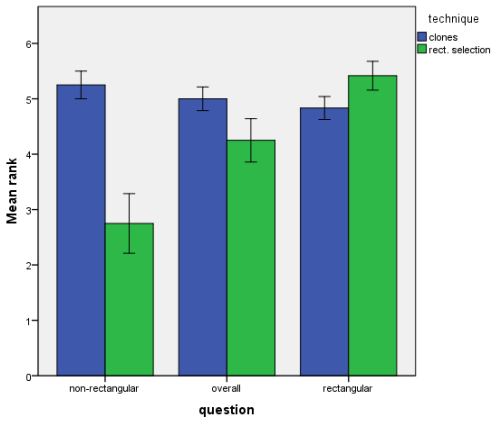

Figure 24. Questionnaire analysis for study 2. Error bars: ± 1 SE.

6 General Discussion

We presented the results of two user studies which investigated creation and editing of clones in mouse-based systems. Our new techniques were shown to be either faster or at least competitive for almost all situations. We want to emphasize at this point that our design targets quick and easy object cloning and editing for simple tasks. In other words, we did not target a user interface that can achieve every possible replication or editing task. One option for that is to extend the dialog based interface, and another to implement an "animation" interface that instatiates objects based on a start and end position, together with an interpolation method. However, these designs involve at least one more step compared to direct cloning, and hence are unlikely to be faster. Moreover, if one targets non-linear interpolation, this necessitates a much more complex user interface, too. In our experience, this also requires that user understand animation concepts.

In summary, we believe that clone editing is an ideal addition to drawing systems, as it works well overall and excels in situations where rectangle selection cannot be used effectively. Seen differently, our results underline the potential for interaction techniques that use the structure of a layout. This is also visible from the fact that rectangle selection performed only well for layouts which were strictly rectangular. Also, it is important to point out that users preferred clone editing overall.

7 Conclusion and Future Work

We presented a new direct cloning interface that is superior to tiled cloning. The method is competitive to smart duplication for grid creations, but the new clone editing interface is more effective in situations where traditional editing methods are problematic to use. In general, participants had a strong preference for the new cloning method, especially for editing tasks in non-rectangular arrangements. Hence, we see our technique as an ideal complement to existing mouse-based drawing user interfaces.

During the evaluation, we disabled creation of overlapping clones, but we are planning to investigate this in future. We also plan to look at user interfaces that enable the user to first create a path or non-axis aligned line, and then to specify the replication of clones along that path, similar to facilities for laying out text along a path. The results from the second user study already provide some positive evidence for this direction. However, any such user interface needs to support editing the path post-hoc, which introduces additional difficulties. We are also planning to investigate the efficiency of the presented pop-up editing interface. One of the ideas is to add another option to "edit all preceding" clones and investigate if users would use that option. A comparison with lasso selection in pen-based systems is also being considered. Last, but not least, we plan to investigate if our new techniques can be combined with existing techniques in drawing systems and how our new techniques can be applied to calendar systems to make editing of repeated events simpler.

References

[1] Bier, E. and Stone, M. Snap-dragging, *SIGGRAPH '86*, 233-240.

[2] Block, F., Villar, N., and Gellersen, H. A malleable physical interface for copying, pasting, and organizing digital clips. *Conference on Tangible and Embedded Interaction 2008*, 117-120.

[3] Chen, J., Bowman, D., Effectiveness of Cloning Techniques for Architectural Virtual Environments, *VR 2006*, 103-110.

[4] Chen, J., Bowman, D.A., Lucas, J.F., & Wingrave, C.A., Interfaces for cloning in immersive virtual environments, *Eurographics Symposium on Virtual Environments 2004*, 91-98.

[5] Coburn, F. D. 2006 *CorelDRAW X3 Unleashed*. Unleashed Productions, Inc.,

[6] Dehmeshki H., Stuerzlinger W., ICE-Lasso: An Enhanced Form Of Lasso Selection, *IEEE Symposium on Human Factors and Ergonomics 2009*, 630-635.

[7] Dehmeshki H., Stuerzlinger W., Intelligent Mouse-based Object Group Selection, *Smart Graphics 2008*, 33-44

[8] Fujima J., Lunzer A., Hornbaek K., Tanaka Y. C3W: Clipping, Connecting and Cloning for the Web, *World Wide Web Conference 2004, Alternate track papers & posters*, 444-445

[9] Fujima J., Lunzer A., Hornbaek K., Tanaka Y. Clip, Connect, Clone: Combining Application Elements to Build Custom Interfaces for Information Access, *UIST 2004*, 175-184

[10] Grossman, T., Baudisch, P., Hinckley, K. Handle Flags: efficient and flexible selections for inking applications. *Graphics Interface 2009*, 167-174.

[11] Masui, T., HyperSnapping. *IEEE 2001 Symposium on Human Centric Computing Languages and Environments*, 188.

[12] Meggs, P. B., 1992 Type and Image: The Language of Graphic Design, Philip B. Meggs, Wiley

[13] Mullet, K. and Sano, D. 1995 Designing Visual Interfaces: Communication Oriented Techniques. Prentice-Hall, Inc.

[14] Neely S., Booth K., Tanner P., The fill interpreter: a unified view of brushing, filling, and compositing, *Graphics Interface '89*, 121-129.

[15] Raisamo, R., Räihä, K. A new direct manipulation technique for aligning objects in drawing programs. *UIST '96*, 157-164

[16] Smith, G., Salzman, T., & Stuerzlinger, W., 3D Scene manipulation with 2D devices and constraints, *Graphics Interface 2001*, 135-142.

[17] Stylos, J., Myers, B. A., and Faulring, A. Citrine: providing intelligent copy-and-paste, *UIST '04*, 185-188.

[18] Williams, R. *The Non-Designer's Design Book*, Third Edition, Peachpit Press, 2008.