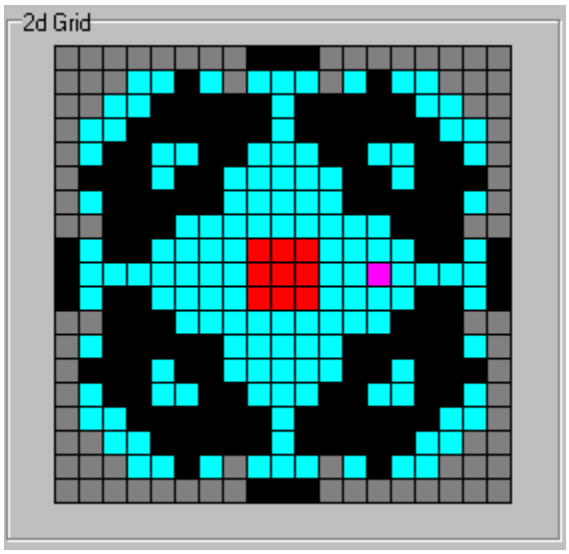
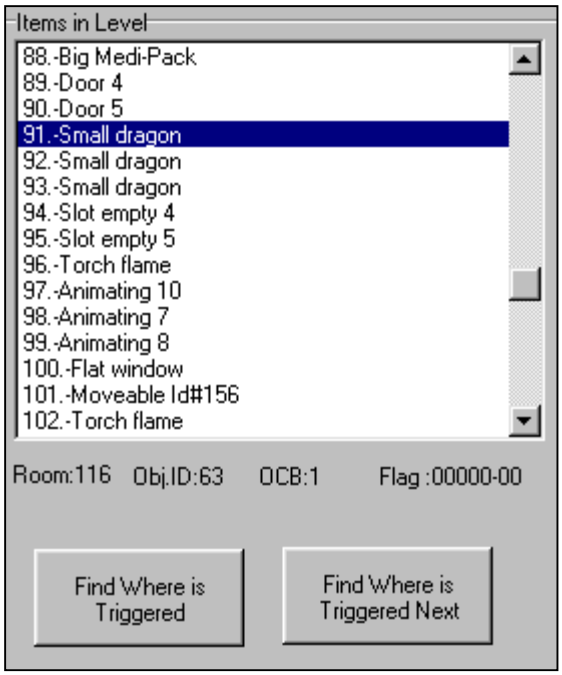
FEXINSPECT 1.0

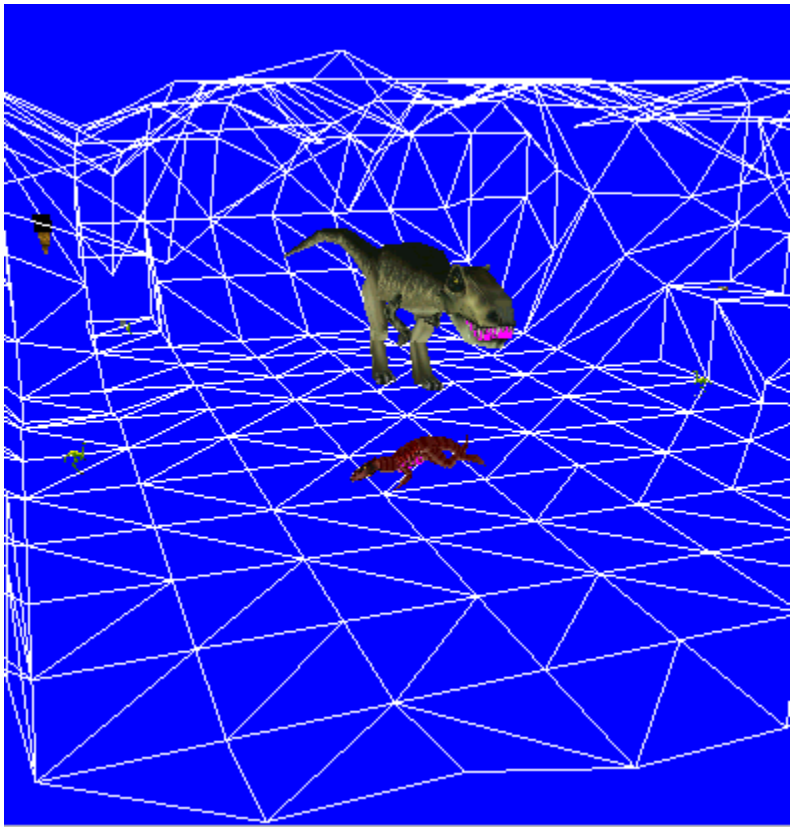
By Turbo Pascal, September 2004.





Overview:

Fexinspect is a program based and inspired on Michiel's Flooredit tool, which allow to open a compiled Tomb Raider level and inspect in detail how it was done, revising the item and triggers data so the user can learn how to setup and implement puzzles and discover how Core design did their levels.

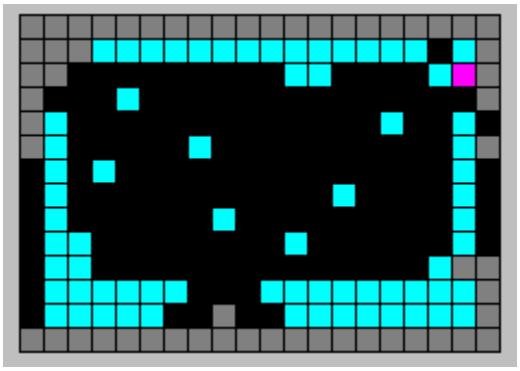


Flooredit is a tool that seem is not going to be updated anymore and I was needing some additional function abilities that allow me to search easily where some commands are implemented and also to inspect some unknown data current not documented in the Tomb Raider file format.

Fexinspect will allow me to continuing hacking the Tomb Raider file format and will allow to the user to learn how some items need to be setuped or for what they are used for. Also, it will help you to continuing playing a custom level when you are stuck, it will tell you where a item is located, where is triggered, and it will show you some areas you could had missed in game.

This Fexinspect version doesn't allow you to modify anything in the opened level, so you will not found any "Save" menu option, the idea is to test what you found making a test room with the editor and the info you found.

It support Phd, Tub, Tr2, tr4 and Trc files formats, I am just inspecting room info, items info, and trigger info, you will not see anything about light, cameras, sinks, etc, there is not much left to check there anyway; static mesh can be seen in the 3d render window, but you will not get much info about them, except the ID when you click on in.

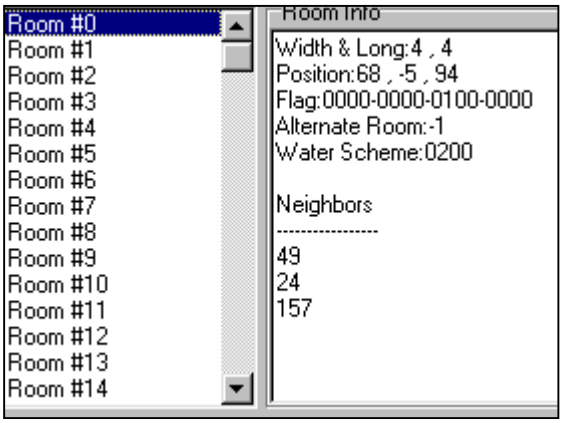


Only one room at time can be seem, please don't bother me requesting to implement a full "show all room" option or show several rooms at same time, I am not doing any Tomb Raider viewer program right now.

Sectors can be selected clicking in the 2d grid system or in the full 3d render windows, if you click a static mesh then you will see in the status bar some info abut it, if you click a item then it get selected and it will show you his properties, if you click any floor/wall/ceiling mesh then the corresponding sectors is selected and the info for it is show.

Inspecting the level room by room.

Open the level you want t inspect from menu “File”, “Open”, the program will load the file and it will take a little time parsing the floor data used in whole level.



You will get the room list numbered from “0”, to the side you see some important info related to the current room:

- Widh & Long room **dimension** in sector units.
- Room **Position** in sectors units, note “Y” coordinate could be negative, which mean going Up, meanwhile positive values mean going down.
- Room **Flag**: 16 bits flags shown using “0” (off) and “1” (on), they are in 4 bits group to easily know each flag position.

These are the current documented Flags info known. (Info taken from Popov web site)

Room Flags:

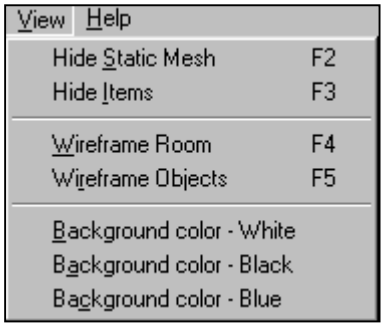
Bit 12	unknown. Not found in any TR levels, but when the 'P' flag is set in the TRLE, this bit is set...
Bit 11	unknown. Not found in any TR levels, but when the 'D' flag is set in the TRLE, this bit is set...
Bit 9	The room has some 'water reflectivity'. TRLE sets this bit when the R ('water reflectivity') option is used (in the same time, the amount of reflectivity typed by the user + 5 is put in the Param1 byte). Unknown effect on the rendering / the game.
Bit 8	In TR3/4: mist room. TRLE sets this bit when the M ('mist')option is used (in the same time, the amount of mist typed by the user is put in the Param1 byte). Unknown effect on the rendering / the game. For tr1,tr2: unknown meaning.
Bit 7	quicksand. TRLE sets this bit when the NL flag is used for the room (in the same time, it sets flag 0x2000 for each vertex of the room above which are at the same floor height than the ceiling height of this room ie. the vertices which are at the joint of both rooms)
Bit 6	Unknown. A lot of rooms have this bit set but it seems it does nothing...
Bit 5	Lara's ponytail gets blown by the wind (outdoor room)
Bit 4	Unknown.
Bit 3	set if the sky can be seen from this room. Used to speed things up: if no rendered room has this bit set, then the sky can never been seen, so it is not rendered. Else, if at least one visible room has this bit set, then the sky must be drawn because it is (could be) visible.
Bit 0	the room is filled with water

- **Water scheme**: here is shown a value in hexadecimal format, like “aabb” where “aa” mean the high byte, which store an **unknown info**, mostly a values 0,1,2,3. “bb” mean the low byte which store the water scheme, reflectivity, mist intensity, etc defined in the room, mostly here are values from 0 to 15 (00 to 0f).
- **Neighbors** are a list of portal to neighbor’s rooms, that way you will knows which are the next room in the link.

Each time you select a room, it is automatically centered if the option **“Automatic”** is checked, or you can center manually the room clicking the **“Center”** labeled button.

You can inspect the room in the 3d render windows, using mouse right button plus shift key, Ctrl key, for pan, zoom, and rotate the room model.

Into the 3d render window you will see the room geometry full textured; ornaments and Item objects in this room are also rendered full textured; Except the current selected item which will be seen in wire frame mode; you can at any time to press “F2” key for toggle Hide/show ornaments; “F3” key for toggle Hide/show Items; “F4” key for render the room meshes in wire frame mode or “F5” key for render ornaments and items in wire frame mode (useful for reorganized objects into the room).
You can from menu “view” change the 3d render window the background color to White, Black, or Blue.



Along with the room info, each time you select a room, it is also shown in the 2d grid the map for current room, where you can see which sectors are walls (gray color), door portals (black color), regular floor sectors (cyan color), trigger sector (magenta color), mortal death sectors (red color) and mine cart /trigger triggerer sectors (green color).

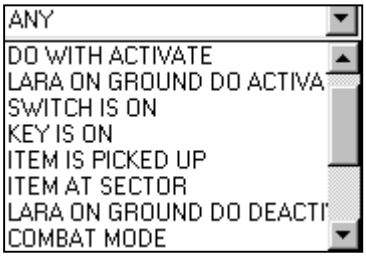
When you click a portal sector then at the “trigger details” window the program will show you to want room this portal is, it could be to the room side, room below or room above. When you click a trigger sector then it will show you a detailed trigger descript with his full commands. The info you will see is:

- Trigger type. Kind trigger used.
- Timer: how much seconds for this trigger.
- Flags, 5 activation bit flags, plus 2 bits that should be always unused.
- If the “only once” option was applied.
- If the trigger type required a trigger item then it is shown.

There is displayed all commands included with trigger and their parameters; note that all items parameters are shown using his full description and his correlative position index in the Item table which can be used for locate the item from the item table and to get its properties.

All Triggers types and commands are shown using DXTRE3D label descriptions, if you are a TRLE user then these are the equivalents:

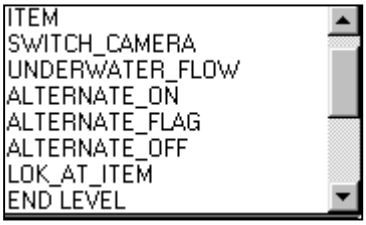
Trigger Types:

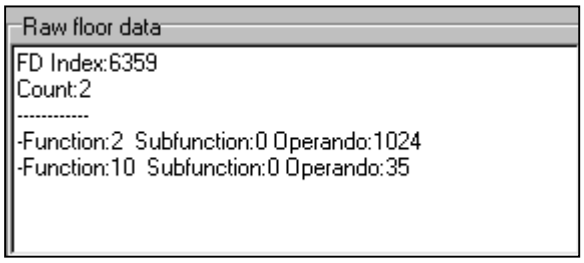


- Do with activate……Trigger,
- Do with deactivate……Antitrigger.
- Lara On ground do with activate……Pad.
- Lara on ground do with deactivate……Anti-pad.
- Switch is On……..Switch,
- Key Is On……Key.
- Item is picked Up……..Pickup.
- Item at sector do with activate……heavy trigger.
- Item at sector do with deactivate.……heavy anti-trigger,
- Trapdoor……Dummy.
- Heavy switch……heavy switch,
- Monkey swing……Monkey.
- Combat Mode……Combat.

Commands:

- Item ……Object
- Switch camera……Target
- Look at item……?
- Underwater_flow…….?
- Alternate ON……Flipmap.
- Alternate_Flag……Flip On.
- Alternate_OFF……Flip Off
- End_Level……....Finish.
- Play_soundtrack…….CD.
- Effects……..Flipeffect.
- Found Secret……..Secret.
- Bodybag……..Bodybag.
- Flayby_camera……..Flyby.
- Cutscene……Cutscene.





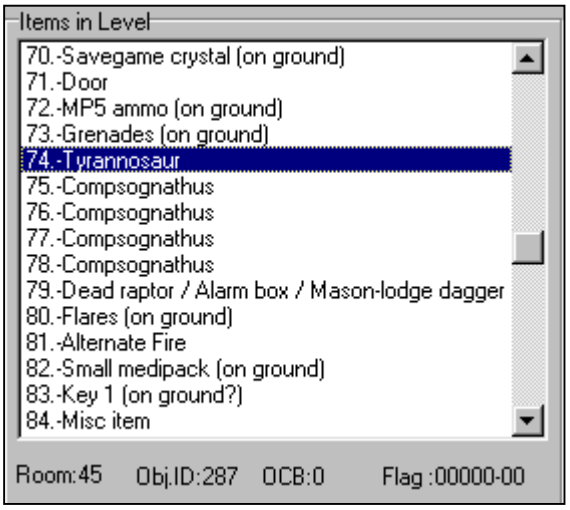
There is also a control labeled “Raw floor data”, it is just an un-detailed floor data shown in “raw” mode, meaning it is not human read parsed, I am using this just in case there are any undocumented floor data found in TRC file format; please ignore the info in this dialog except if you are a expert parsing tomb raider floor data info.

Table Items:

The control labeled “Items in level” is the list all items placed in the level, they have his full description and his correlative number which match the one used on the Item(X) command for activating that trigger.

When you click a item in the 3d render window, the program automatically will search and select the item from this table, so you can inspect his properties:

- Room: where this item is placed.
- Obj. ID. Internal id used by this movable.
- OCB. Object Code Bits for Tr4 and trc levels.
- Flag. 5 activation bits plus “clear body” and “invisible”.

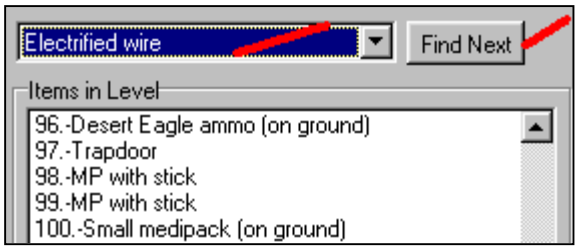


When you select manually an item from this table, you can do double click on in for tell the program to search and display the room where it is located.

Knowing how an item is used:

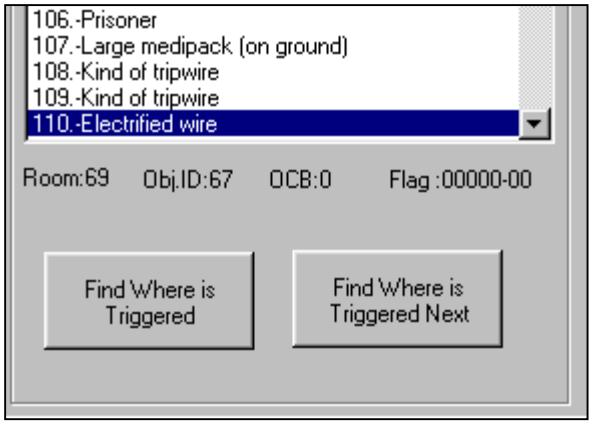
When you are in the editor you can see the Movable list descript available for your opened level base, some items could have unknown meaning for you, and you are wondering for what they can be used or how they are implemented. One way to discover that is inspecting an original level done by core design where this item was used; checking where and how is placed, with what properties and how is triggered and reproducing the same process in a test room using the editor could help you to discover the magic behind this item.

Fexinspect allow you to easily search and found that processes **with not need** to check manually every room and trigger.

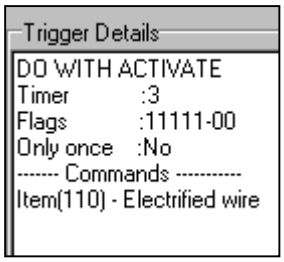


Above the Item table, you found a Combo box control where you can select from all available movables the one you are interested on inspect. As soon you select the movable from this control, the program automatically will find the first instance into the table item, and will locate and display the room where this instance was placed; then you can check the properties, ocb, flags, etc, and how it was placed in the room.

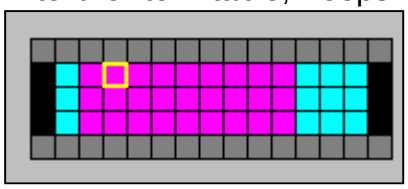
Not necessarily the item is triggered in the same room where it is placed, after inspecting the item in the room, you can click the button “Find where it is triggered”, the program automatically will find & display the room where this item instance is triggered, the sector where the trigger is placed is also selected and a full detail description for the whole trigger is show.



This item instance could be triggered more that once in the same room or in another room using another trigger, so you can click the “Find where it is triggered next” button for tell the program to find and display where else it is triggered. Keep pressing that button until you found all places where the instance is triggered, inspecting all of them will show you if there are different options to be used for trigger this movable.



There could be several instances of the movable into the item table, so you can click the “Find Next” button located near to the movable combo box control for to tell the program to find the next instance of this movable into the item table, keeps pressing the “Find Next” button until you get the message that there is not more instance for that movable, for each instance found by the program, you can check where and how is triggered so you will have a full info inspect how this movable is used in the level.



Getting help when you are stuck in a level:

Using the same process as above you can obtain some hints on what to do next when you are stuck in a level; I was playing a custom tr4 level and as usual I got stuck, after revising all accessible areas, I don't found what to do, there was a closed door than prevent continuing, so I booted Fexinspect and revised the room areas accessible, turning the room in wire frame mode help me to see if there are items that could had missed, selecting the closed door and doing "Find where is triggered" button, will show me if the trigger is in a accessible area, if not I can inspect how to access that area, etc. once I had found what I have missed, I can continue playing the level normally until I get totally stuck again.

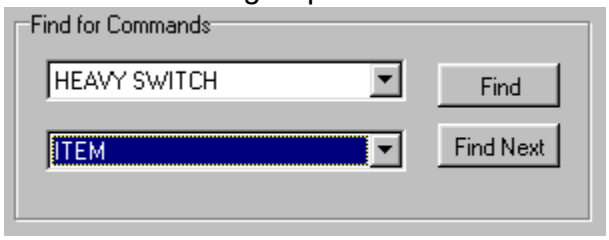
When using Fexinspect you have a clear view of the room, you found where are the portals, if there are so hidden access, where are the keys, where are the triggers, even you can tell fexinspect to search and found where a secret found command is used.

I think you can search for Lara object instance in the level and starting from there you can trace and write a walk trough of the level, revising the neighbors room, checking which triggers you will found first, etc.

Search for a specific trigger or command.

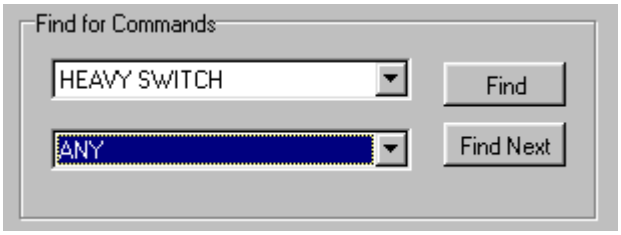
I also added to fexinspect the option to search for a specific trigger or command trough the floor data and will show you the room, sector and the parsed trigger where was used; this help a lot for inspect for example how is used the bodybag command or the heavy switch trigger, instead searching manually for those, you tell the program to find all of them for you.

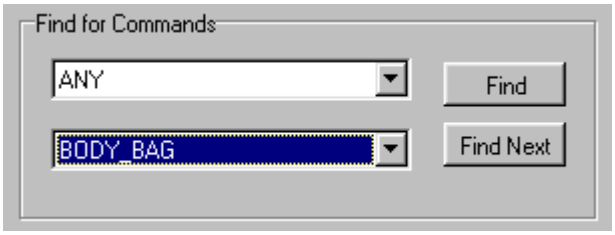
You will found a group labeled "Find for commands", the first combo box controls contains the trigger list types, the second combo box control contains the command list, select the trigger type and the command you want to search, click the "Find" button; Fexinspect will search for that trigger type which also has that command included and it will select the room and the sector where the first occurrence was found, the full trigger will be seen full parsed.



Click the button "Find Next" for tell the program to find the next occurrence of your criteria; you can keep using "find next" until the program tell you that no more occurrence was found.

If you are interested to found just the trigger type no matter which commands it has, then you can select the special option "ANY" in the command list combo box.





If you are interested to found just the command no matter in which trigger type is was used then select the special option "ANY" in the trigger list combo box.

THANKS:

Thank you very much for those who contributed discovering/parsing the floor data file format:

- Yuri Zivago
- E. Popov.
- Michiel.
- Roy Goldbould.

Special thanks to Michiel who inspired me to code this tool.

Turbo Pascal. 23-September-2004.